3-30-2017

Design and Control of a Dynamic and Autonomous Trackless Vehicle Using Onboard and Environmental Sensors

Scott R. Jagolinzer
Florida International University, sjago001@fiu.edu

DOI: 10.25148/etd.FIDC001803

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

DESIGN AND CONTROL OF A DYNAMIC AND AUTONOUMOUS TRACKLESS VEHICLE USING ONBOARD AND ENVIRONMENTAL SENSORS

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCINECE

in

MECHANICAL ENGINEERING

by

Scott Jagolinzer

2017

To: Interim Dean Ranu Jung
College of Engineering and Computing

This thesis, written by Scott Jagolinzer, and entitled Design and Control of a Dynamic and Autonomous Trackless Vehicle using Onboard and Environmental Sensors, having been approved in respect to style and intellectual content, is referred to you for judgement.

We have read this thesis and recommend that it be approved.

_____
Ibrahim Tansel

_____
Yiding Cao

_____
Sabri Tosunoglu, Major Professor

Date of Defense: March 30, 2017

The thesis of Scott Jagolinzer is approved.

_____
Interim Dean Ranu Jung
College of Engineering and Computing

_____
Andres G. Gil
Vice President for Research and Economic Development
and Dean of the University Graduate School

Florida International University, 2017

ABSTRACT OF THE THESIS

DESIGN AND CONTROL OF A DYNAMIC AND AUTONOUMOUS TRACKLESS VEHICLE USING ONBOARD AND ENVIRONMENTAL SENSORS

by

Scott Jagolinzer

Florida International University, 2017

Miami, Florida

Professor Sabri Tosunoglu, Major Professor

This thesis explores the current state of automated guided vehicles (AGVs), sensors, control systems, and wireless technology. The goal of this research is to develop improvements for both vehicle and controller design that allows for improved safety and efficiency. Right now, the main issues are maneuverability of vehicles and control systems being adaptive enough to deal with connection issues between systems. While prolonged connection issues will result in a stoppage of operation of any system that relies on wireless communication, intermittent issues can also cause systems to have an emergency stop. A solution is to offload tasks from the central system and allow the vehicles themselves to have more computational privileges such that they can operate in a semi-independent manner. The result is a system that remedies or limits negative effects that currently cause issues with trackless vehicles and control systems working with remote systems that communicate via wireless means.

TABLE OF CONTENTS

Introduction

Automated guided vehicles began simply as a vehicle following a radio signal emitted from a wire on the ground. This has advanced to the point where we are very close to passenger vehicles being able to drive themselves on a widespread level. Vehicles are used in a wide variety of applications from personal transportation, to carrying payloads, to entertainment purposes. Industrial applications for autonomous trackless vehicles are tough because the vehicles have to be able to operate in a very dynamic and hectic environment that doesn't have the benefit of infrastructure and ruleset that autonomous vehicles on a road are able to use. The fact that the applications can have the vehicles operating either indoors or outdoors or even moving between the two creates tracking challenges for the system.

Controllers that operate autonomous vehicles have to either be contained fully on the vehicle, as with self-driving cars, or can be centrally located and connect wireless to a plurality of vehicles. While trackless vehicles that connect to a wireless control system still need to have on-board computing capability to process the signals they get from the central controller, they are usually tethered to the signal and rely on having a stable connection. There has to be ways to improve the balance of computing capability between autonomous systems that rely on a central control system that allow for more forgiving operation in situations where signal interference is possible.

After work began on this thesis, I worked as the leader of FIU's team for a competition called the NASA Swarmathon and my experience in this competition influenced me and was rolled into this work as both deal with control of wireless, trackless

autonomous vehicles. The NASA Swarmathon competition requires creating a swarm control system that allows multiple rovers of a fixed design to perform a specific task. The task to be completed was to collect and return resources in an arena. This thesis covers work for the competition for both 2016 and 2017. The work on the Swarmathon competition helped inform my trackless vehicle controller design and provided real world results on the capabilities and limitations of existing technology that autonomous vehicles are equipped with.

Literature Review

History of AGVs

Automated Guided Vehicles (AGVs) began use in the 1950's in the form of a tow truck that followed a wire in the floor. This concept advanced to include many types of vehicles, mostly in an industrial setting. The AGVs can range from small forklifts moving pallets in a warehouse to larger vehicles that move shipping containers around a port.

There are various methods for guiding the vehicles. These include wired navigation, guide tape, laser target navigation, inertial navigation, natural feature navigation, vision guidance, or geoguidance.

Wired navigation is simply a wire carrying a radio signal embedded into the path the AGV is to traverse. The AGV has a sensor mounted below it that hovers just off the surface and picks up the signal from the wire, which allows for the vehicle to sense and follow the wire.

Guide tape is similar in principle to the guide wire but the guide tape carries no signal. Instead either a colored or magnetic tape is placed along the desired path and either a vision system or magnetic sensor read and follow the path laid out by the tape. There are advantages and disadvantages to both visible and magnetic tape. Visible tape, while it is cheap to install and make path adjustments, since it needs to be on the surface of the path to be viewed by the AGV, it can be damaged or dirty through daily vehicle and personnel activity. A system that uses a visible guideline for steering control has two ways it handles the guideline to control the steering. The first way, when the vehicle is moving at a slower speed it reacts to the error values of the vehicle with respect to the line. When moving faster, the controller uses the slope of the line to determine the required steering as waiting

for error results could cause the vehicle to not be able to react in time at higher speeds. [J. Lee] The magnetic tape allows for it to be embedded into the path and therefore is not exposed to wear. However, having to carve out a channel and cover the buried magnetic tape results in a costlier install and makes it harder to change the path layout.

Laser target navigation uses a rotating laser emitter and sensor in tandem for distance detection that can be used for position tracking. The laser is reflected off reflective tags that are affixed to walls, poles, or other fixed obstacles, each of which corresponds to a known point in the workspace by the system. Therefore, by reflecting off the various reflective tags, the vehicle can triangulate its position and use that knowledge to plan and execute its motion based on the data that is collected and processed in real time. There are two types of laser sensors, modulated and pulsed. Modulated lasers have a larger range and higher accuracy over a pulsed laser with an angular resolution of 0.006°. Pulsed lasers have an angular resolution of 0.2° and must be interpolated by the intensity of the reflection at each data point to reliably get an accurate location. For the system to work the vehicle needs to be in view of at least 3 environmental reflectors at all times to correctly triangulate its position. [AIM]

Inertial navigation uses a gyroscope that can, within an inch, detect the movement of the vehicle. It also uses transponders located in the workspace floor to verify the position of the vehicle. This system is flexible in that it can be used in a wide range of environments with or without the feedback. [Egemin]

Natural feature navigation uses range detecting sensors in conjunction with gyroscopes to read the layout around it and the vehicle triangulates its position using the natural features on the environment. It uses the data of its surrounding to develop the

shortest possible distance to its destination since the system doesn't necessarily know the layout of its surroundings. [Egemin]

If the natural feature navigation did have full knowledge of its surroundings, then this system would be considered a geoguidance system as it operates in the same way a laser guidance system works but without the reflective tags.

Vision guidance uses cameras to view their surroundings in a full 3D image and then builds a 3D map of the workspace. It can then match up its live view with that of its 3D map to be able to determine where it is in the workspace. It can also be programmed to recognize certain features such as obstacles or humans as to allow for the vehicle to stop or reroute. [Han]

Future Uses

AGV and other autonomous vehicles are becoming increasingly prevalent as technology and automation have advanced. From simple machines following a wire to a squadron of drones flying a coordinated mission, autonomous vehicles are the future and are proliferating into every industry.

As technology advances, there is a push for automation as it leads to increased efficiency and consistency in costs and output. While most of the automation in industry has been for machines that work in a stationary environment or at least attached to a track that can guide its movement. Transport of material, especially within an industrial environment is something that can be automated, but there are challenges due to the dynamic nature that they can potentially encounter. This results in the need for complex vehicles that operate with robust control systems that can not only manage a fleet of

vehicles, but monitor the environment and operate with safety in mind without compromising efficiency.

Design of AGV

Steering

Steering is one of the most important factors in the AGV because the method of steering has an effect on the mobility and control of the platform. There are three majors steering schemes used on AGVs.

- Forward and backward movement
- Works well with tugger AGVs
- Turning limitations
- Works well in looped systems

Figure 1: Differential drive with layout highlights. [Transbotics]

The most basic steering control is from a differential drive wheel configuration[1]. This is when there are two drive wheels that are parallel to one another and the turning process is created when there is a difference in wheel speeds. This is similar to how a tank steers. It benefits from its design simplicity but is unable to perform more complex movements.

Another steering method is a three-wheel steer drive configuration. There are two castor wheels and the one powered drive wheel that can also spin to control direction. This is the type of steering that is

- Used on most fork lift AGVs
- Very smooth and accurate motions
- Drives like a tricycle vehicle
- 360° movement in any direction
- Develops the guide path to compensate for the over swing created by a single-steer vehicle

Figure 2: 2 Steer-drive with layout highlights. [Transbotics]

[1] Images for wheel configurations from https://www.transbotics.com/learning-center/drive-steering

- 360° movement
- Tight turning capability
- Allows the vehicle to move in most directions
- Vehicle movement allows for loading and unloading in most areas

Figure 3:Two steer-drive with layout highlights. [Transbotics]

available on most forklifts. This method is more maneuverable than differential steering and can perform smooth and accurate motions but still isn't the most maneuverable option.

The third option is a two steer-drive wheel configuration. This configuration has two drive wheels that are able to be rotated and two castor wheels. This steering configuration allows from the vehicle to move in any direction and turn on the spot. Similar to the 2 steer-drive wheel configuration is the Quad wheel configuration. This combines two fully rotatable drive wheels with two castor wheels to give the best range of motion and highest maneuverability. [Transbotics]

A possible improvement that will be explored is having four drive motors that can all be rotated independently. This has the advantage over the two steer-drive configurations in that all four wheels are driven, allowing for double the available power and can also allow for power efficiency and reliability by disengaging a pair of wheels to drop the power consumption by half or allow for continued operation if a motor for a wheel breaks

- Allows the vehicle to move in any direction
- Center point of the vehicle always stays in the center of the guide path
- Highest degree of maneuverability
- 360° movement in any direction
- Works well in areas where space constraints may be a concern
- Recommended for use with laser navigation technology

Figure 4: Quad drive with layout highlights. [Transbotics]

or a sensor malfunctions. This fault-tolerance system will allow for increased efficiency and reliability.

Load Capabilities

Automated vehicles will always be dealing with a load that it will be transporting whether it be people, cargo containers, or other items. The vehicles need to be designed to properly be able to handle expected loads while performing the tasks that will be required of it. Because of this, AGVs are usually specially designed for the job they will be performed.

There are any ways that the loading capabilities can be affected. The size and weight of the cargo is a big factor. Heavy items will need a stronger vehicles and drive motors. Larger items will require either a larger vehicle that can fit the cargo or can work in tandem with another vehicle to move an oversized object. Also, the desired capacity of the vehicle will affect the design. While the cargo can be small if a large volume is needed to be moved, cost analysis can be done to see if it is more cost effective to operate a larger number of smaller vehicles or a smaller number of larger vehicles. The size constraints of the workspace are also a factor that could potentially impose constraints on the design. [Vis][Kaloutsakis]

While individually each of these factors have an easy solution, when all the above need to be taken into consideration simultaneously, with even more factors not related to the load such as cost to build, cost to operate, and time constraints.

Multi-axis Articulation

While the basis for the AGV and the autonomous trackless vehicle being proposed is the translational motion being performed by the base to move in a desired path, further

articulation can be performed upon the base platform to further augment the motion (as in an entertainment application) or to further increase the workspace and flexibility of the system (as in an industrial application).

While the feature of multi-axis articulation is not a requirement for an AGV it is worth exploring as to possible uses that can improve the efficiency and value of the AGV. There are various options to increasing the articulation of the vehicle.

Roll, pitch, and yaw rotation can be implemented under the platform that holds the load being placed onto the vehicle. These added articulations can eliminate possible machinery and actions from a work environment using an AGV without such articulation. The yaw rotation can be used to rotate cargo such that it is facing the desired direction not only during transport but also for when it is being moved onto or off of the vehicle. This would remove then need for the vehicle or machinery moving the load from having to reposition itself. Roll and pitch rotations allow for the platform holding the load to tilt with respect to the floor. In an industrial setting this can allow for the vehicle to drop of its load on its own. In an entertainment setting these articulations can be used to impart forces onto the rider simulation and augmenting the action that is occurring to them. [Lee]

A possible way to control the articulation of any multi-DOF system on top of the base platform could by having two motors work together for each joint where they apply forces opposite each other to create controllable tension and stiffness in the system. This mimics how muscles in the human body act upon the joints of the skeleton. Implementing a stiffness control system into the platform can allow better control and handling of cargo, especially when such cargo has significant mass or needs to be held in specific positions over a prolonged period. [Koganezawa]

An elevator system can allow for the platform on the AGV to move in the vertical direction allowing it to raise its platform up to a desired level for it to accept its load. An example of where this could be useful is where packages are moved along conveyor belts at different heights and this can allow for the vehicle to position its platform at each level necessary to allow for the packages to slide onto the vehicle.

A horizontal slide can be added to allow for the vehicle to extend its reach in areas where the base of the vehicle is constrained. This could be used in a port where instead of a crane system attempting to center the container on the AGV, the crane system does the best it can and the vehicle can precisely center the platform under the cargo as it is lowered.

All of the above articulations can be combined as needed to allow for the platform to become a useful tool that can replace some machines and actions and allow for other tasks to become quicker and more efficient. The more functionality that can be added to the autonomous vehicle, the more complex the operations that the vehicle can carry out.

Control Systems

Current Control Systems

In controlling AGVs there are two types of control, offline and online control, with the latter having two sub categories of a centralized or a decentralized system [Vis].

Offline control is when the control scheme for a system is all preprogrammed and there is no form of feedback. This control is best used for conditions when all aspects of control can be accounted for ahead of time since there is no form of feedback. All parameters must be accurately known beforehand and remain constant due to the lack of feedback, and therefore an inability to compensate its routine. While this form of control

could be used in a highly-controlled situation, the lack of feedback and adaptability render it useless in a dynamic industrial setting, especially in terms of safety.

Online control is used in situations where the control system needs to be capable to make decisions in real time. A decentralized system is one in which either the controller is on board of each vehicle and they can talk to each other or from multiple decentralized control systems that are in place at multiple places along the controlled process. A centralized system has one system that monitors and controls all the vehicles. A centralized system frequently gathers data from all of the vehicles and sensors and uses that data to coordinate the network. [Vis]

A hybrid control system consisting of a centralized and decentralized system has already been explored in some capacity [Arora]. It proposed a three-layered approach to plan, execute, and regulate the commands executed by the controller. However, this proposed hybrid controller has some limiting factors such as vehicles being unable to go backwards, fixed workspace, same size workstations, and only two jobs being processed at the same workstation at one time. While there was a time where this controller provided a suitable solution for hybrid control, today's more dynamic vehicles and environments require a more robust and flexible controller that can be adaptable and reliable.

Zone control is when the vehicle workspace is sectioned off into various zones. Only one vehicle is allowed to be in a zone at any one time therefore avoiding possible collisions. It's a simple solution to managing the movement of autonomous vehicles but has some disadvantages. These zones have to be of a size that is much larger of the vehicle and there needs to be an adjacent zone to the vehicle that is empty for it to be able to move out of one zone into another zone. These requirements take up a lot of space and also waste

space by requiring empty zones or else the system would become gridlocked. While this can work if properly implemented on a macro level, the movement of the vehicle within its zone still needs to be managed. [Fanti]

Sensor control uses an array of on board and environmental sensors that detect other vehicles and environmental objects. This allows for the control system to move freely and react when a sensor picks up feedback. This form of control is less restrictive but relies on the accuracy of the sensors to function properly. [Payton]

Combining zone control with sensor control allows for the objects to read their environment and also adds another layer of security zoning vehicles such that they have a cushion where they know they can safely operate.

Wireless Communication

Wireless communication is a great tool, especially in automation, that allows for systems to communicate across a distance or in an environment that a wired connection would be unfeasible. However, wireless communication has the possibility of having the signal quality degrade over a distance or though objects or structures. This can cause a complete loss of signal or for parts of the signal to get lost or corrupted. In this case a system that requires communication via a wireless system would have to make an emergency stop as its operation is linked with the system it is wireless communicating with. This causes unwanted downtime as the system would need to be brought back online, even if there was only a very brief loss of connection.

There are some ways to improve the reliability of the wireless signal. [Mahmood] When a wireless signal is broadcast from one terminal to another, the signal can be disrupted by signal congestion at a certain node, transmission errors, signal interference,

or from a node failure. Regardless of the cause of the lost signal, a way to improve the signal reliability is through retransmission or redundancy. Each of these methods involve resending the data that is lost, but differ in their approach to how the data is resent. To break down the difference between retransmission and redundancy, you must understand how wireless information is transmitted. Information is send in groups of data called packets. When there is a problem with the signal it can be a result of entire packet loss, as would be more common with signal interference or node failure, or it can be a result of data within the packet being missing or corrupt, as would be more likely with transmission errors or signals getting mixed or scrambled.

Retransmission would resend an entire packet if it does get a conformation of reception of the full packet, even if there was only part of the information missing from the packet. Redundancy will only resend the missing or incorrect data. While that seems like it would be the obvious choice as it would require less data to be resent, there is an inherent upfront cost of extra information that would need to be put into each packet that would allow for them to reconstructed from smaller parts. There is some crossover point in overall data requirements of the system between retransmission, using more bandwidth to resend a lost packet, and redundancy, using more upfront bandwidth and less bandwidth when correcting an error, such that a more reliable network should use retransmission and a more unreliable network should use redundancy. There could be other factors such as time sensitivity that could be benefited by redundancy assuming there is enough bandwidth to handle the inherent increase in the signal size due to the embedded information.

In an environment where a wireless node cannot be hard-lined to the system, that would require multiple wireless nodes to pass the signal along. This increases the

possibility of error as any of the issues can occur each time the signal is transmitted or received at each node. [Mahmood]

Sensors

The sensors that will be used on board and in the environment, will play a large role in the capabilities of the sensors. While the robustness of the control scheme is the foundation for autonomous control, an equally robust array of sensors is required to feed enough pertinent information that the system can make well informed decisions.

Vision sensors are those that use the visible light spectrum to capture an image for the system to process. Since a computer cannot inherently look at a picture and understand what it is displaying like a human can, the controller needs to be programmed to identify pertinent information from the video feed it is receiving. This can be as simply as recognizing a colored line on the floor and knowing it should follow it to recognizing certain shapes such as walls, other vehicles, or any other obstacles it could come across and knowing how to take evasive action. A vision system is very flexible but requires extensive programming work to get the most out of it. [Lee]

Ultrasonic sensors can typically be used as proximity sensors since it is very easy to time the bounce of the signal and determine the distance of the object reflecting the sound waves. While these sensors would not be used for controlling the vehicle, they are instead used as a fallback safety system to warn and allow for the vehicle to stop if anything unexpected ever enters a predefined workspace of the vehicle. When dealing with large and expensive machinery, especially machinery that is moving autonomously, it is important to have multiple sensing and safety layers. [Josef]

Infrared sensors can be used for tracking of the vehicle and positional sensing. While similar to the vision system, since these sensors only operate on the infrared wavelength, special infrared emitters and reflectors can be placed on the vehicles and in the environment to allow for positional data to be determined since the environmental emitters and sensors will always have a fixed and known location. [Nanda]

Radio frequency sensor picks up radio signals that are present. In the history of the AGV, this sensor has been used to allow the vehicle to read and follow a wire that emits a radio frequency that is embedded into the floor that defines the desired path of the vehicle.

Magnetic sensors pick up on a magnetic field that is present. A magnetic sensor can be used in the same way that the radio sensor was used by reading the signal from a magnetically charged wire that defines a path for the vehicle.

Gyroscopic/force sensors can measure the forces that the AGV is experiencing due to acceleration. This can be used to gauge and control the acceleration and deceleration of the vehicle while also monitoring lateral forces that could be imparted by taking turns with a moderate speed.

Global Positioning Systems are another way to detect the position of the vehicle and is especially useful in an outdoor work environment that is exposed to more variables such as weather and time of day that can affect some of the sensors mentioned above.

All the above sensors can be used together to give an automated vehicle a plethora of instruments that allow for the vehicle to read and navigate its environment. [Yuan] The sensors each have their strengths and weaknesses that must be considered when choosing the right sensors to design a control system around. It is also important to have built in fail safes and redundancies into the system such that control will not be lost should a sensor go

down or an environmental condition affect the system. One example is if the lights go out in the workspace and the controller only operated off of vision sensors. This would cause the system to lose all input from the sensors. And while at that point one would hope that the controller is programmed to shut down in that scenario, there is a moment while all these vehicles would be trying to stop that they would be blind and at risk for collision. In this scenario ultrasound and infrared sensors can be used to protect against this situation. Infrared sensors are not impaired by the lack of light and use infrared emitters placed n the environment to determine its position. Ultrasound sensors rely only on sound waves to detect the proximity of objects from it. This could allow for the vehicle to detect any objects that would be in its vicinity and allow for it to maneuver in a way that it would be able to avoid the object although it is not able to 'see' it in the conventional sense.

The infrared sensor and emitter clusters depicted below were developed for use on indoor flying robots. [Roberts] However, the sensor design can be used with an autonomous vehicle control system. The design calls for a spherical array of infrared emitters and sensors. In the figure below they are depicted by the blue and red spheres respectively, however they are only depicted as two distinct objects to clearly show the interaction between the emitting and sensing portions of two sensors. The actual sensor is a sphere with both the emitters and sensors interspersed between each other. This allows for each vehicle outfitted with the sensor to not only read its environment but to also be able to be sensed by other vehicles and also by sensors in the environment that monitor the positions of the vehicles.

When the sensor picks up a signal, it measures the intensity of that signal at each of the sensor nodes. It uses the three strongest signals and then triangulates the direction and uses the signal intensity to determine the distance the object emitting the signal is from the sensing object. This can be done since the geometry of the sensor sphere is known and the decay of the infrared light intensity can be experimentally found and then applied to calculating proximity.

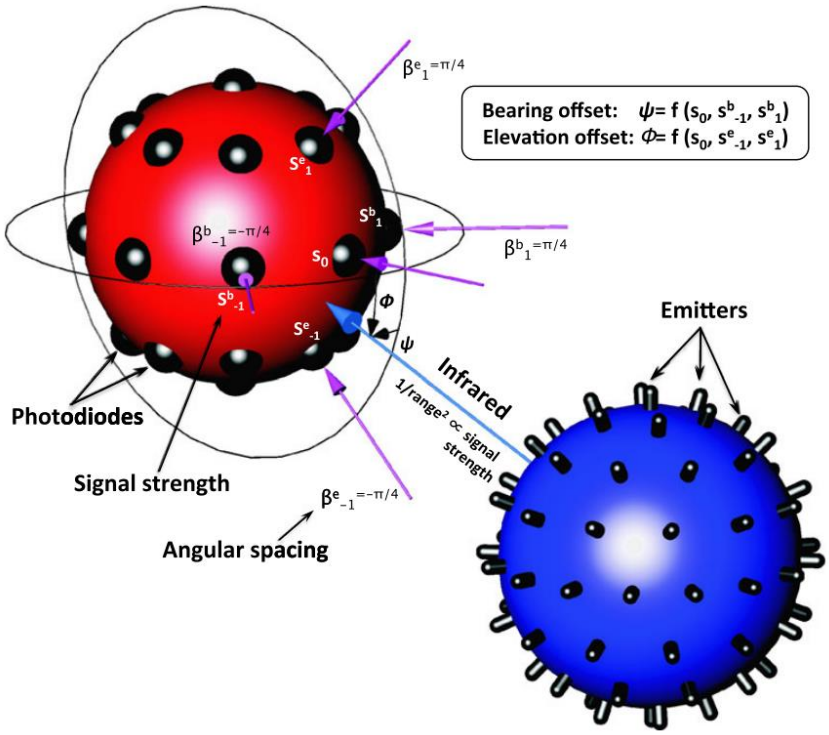

Figure 5: Diagram of 3-D relative positioning sensor for indoor flying robots [Roberts]

Path Planning and Layout Development

Deadlock Avoidance [Fanti]

With the maneuverability of the vehicles deadlocks should be easy to avoid. The path planning is to be designed that there will be right of ways and minimize areas where designated paths would cross. Knowing the position, velocity, origin point, and destination point, the paths can be made such that vehicles can be manipulate in a way that they never arrive at a crossing point at the same time.

Ant Algorithms [Xing]

Ant colony optimization can be applied to path planning for autonomous vehicles. Ant colony optimization the paths taken by the system are generated randomly and tested. Each path leaves behind a trail or pheromone that can be read by the ants that follow in its footsteps. The pheromones strength declines over time to the paths that see more traffic leave behind a stronger pheromone trail. The shorter the path the more traffic it will see thus alerting the following 'ants' to this optimal path. In this system each vehicle is analogous to an ant and as it travels between two destinations the path will be evaluated on its strength and leave behind its pheromone for vehicle to follow in its path. As more iterations of the path are taken the optimal path is recognized. Ant colony optimization is also adaptable as if there is ever a change in path or obstacle encountered the process will be repeated to again develop the most efficient path between the vehicles destinations.

Backstepping [Xu]

While planning path profiles for AGVs, all known obstacles are taken into account like walls and the predictable motion of other planned paths. The controller can then use this known data of the workspace layout and develop the most efficient path for the vehicle to carry out its task. However, not everything is known or predictable ahead of time. There are many unknown objects that can end up in the planned path of an AGV and the controller's ability to recognize and react to the obstacle is very important. The most common cause of accidents is by collision of AGVs with objects in their environment, either known or unknown. To avoid this the controller needs to be able to recognize an obstacle early enough and then have the proper programming to successfully avoid the obstacle. To do this path re-planning is required.

One method to re-plan the path is by using back-stepping. Instead of scrapping the whole path when an obstacle is encountered the controller reroutes by calculating the most efficient path from the point before the obstacle taking into account the new obstacle. This method of path re-planning allows for the controller to react on the fly and is flexible to many types of obstacles and situations.

Methodology

NASA Swarmathon

The NASA Swarmathon is a nation competition between Universities, held annually at Kennedy Space Center, to develop swarm search methods for exploring the surface of other planes and collecting potential resources. NASA's goal is to coordinate the use of a multitude of cheaper rovers versus sending a limited number of very expensive rovers. By having more rovers work together there is less risk of having something fail and crippling the entire mission. Having schools compete is a way to spur new ideas that can be used or built upon for NASA space missions.

2016 Competition

Competition Goals and Rover Capabilities

The goal of the competition is to collect as many resources as possible and return them to a designated zone and drop them off. This task is carried out by multiple rovers, 3 in the preliminary round and 6 in the final rounds, that can identify apriltags that represent resources and then return them to a collection zone that is located in the center of the arena and is represented by apriltags as well. Since the tags are only digital representations of resources, each tag is unique and once "picked up" it is registered with the main system so that other rovers do not try to collect a tag already collected.

The rovers themselves have an onboard computer and various sensors. The sensors include a vision camera for detecting apriltags, three front facing ultrasound sensors for detecting obstacles, and a compass, gyroscope, and GPS to track its location. Using the sensors and rovers as provided, a search algorithm had to be developed to collect the tags in as reliable and efficient of a manner as possible.

The program is identical for all the rovers so it must be generalized enough such that all the rovers don't just follow each other around mimicking each other's movements. The program should also be adaptable enough that it works regardless of arena size or number of rovers that are working together. The rovers are also not allowed to directly communicate but instead all line up wirelessly to a central host computer that can then relay information back out to the other rovers and keep track of all the rovers progress via a graphical user interface.

Swarm Algorithm Development

The rover system runs off a program called ROS, which is short for Robot Operating Software. It is an open-source software that runs on Linux. The rovers came preloaded with a basic framework and sample code that had the rover picking a random compass heading, turning to that heading, then moving 1 meter in a straight line, and repeating until a tag was found. Using its location tracking abilities, once a tag was found it turned towards the origin of its coordinate system, which represents the center drop off location for the arena and moves back to (0,0). Having never used ROS or Linux before, this sample code was immensely helpful in figuring out how ROS worked. The code is written in C++ but there are ROS specific commands, libraries, and shortcuts that are used for such tasks of communication between parts of the code, calculations used for positional tracking, and other calculation based tasks.

The best ideas that were developed in brainstorming were a spiral search, box search, out and back, and rake search. Each method had positives and negatives with the rovers available although some did not arrive until we were able to physically test the code on the test rovers provided by NASA.

The spiral search method is simple in principle, the rovers search over an area by starting in the middle of the desired area and then moving in a spiral pattern that is compact enough that no area is missed while minimizing overlap of area searched. It also has the benefit of having the rover constantly moving whereas all the other search methods have the rover either rotating in place or moving in a straight line. This method was easy to program as the spirals were created by simply combining a linear velocity component with a rotational velocity component to create the desired spiral shape. The limiting factors for those values were the speed at which the rover would be unable to recognize the apriltags and the corresponding rotational value that allows the spiral to be the right size to maximize the efficiency of the search pattern. In simulation, this method worked exactly as envisioned. The rover went to a desired location, began its spiral, and upon finding a tag, returned in a straight line to the center and returned to the previous spot of the last tag found, repeating the process. However, the problem arose when testing in on the physical platforms. The way the rovers are set up the computer running ROS interfaces with an Arduino board that interfaces with the sensors and motors. The way the commands are passed causes the physical rovers to only perform one kind of motion at a time, either linear or rotational. The issue was believed to be due to how the commands are passed between systems, but since the code must be able to work on the rovers as is and could not make

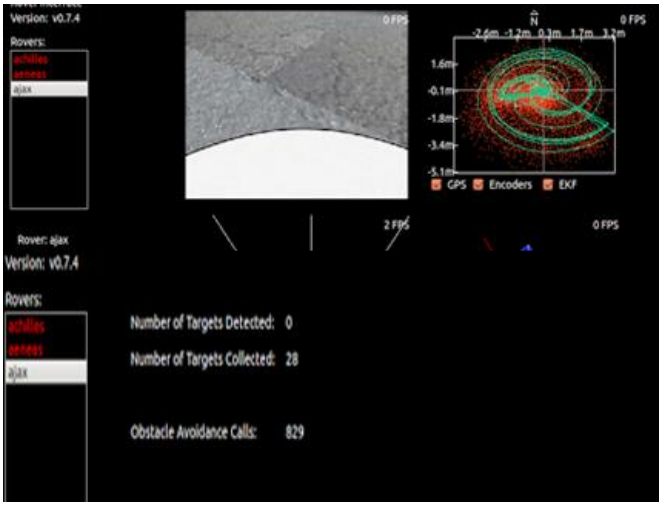

Figure 6: Simulation results for the spiral search method

changes to how the Arduino board handles and outputs signals, this search method was abandoned.

The box search algorithm was an evolution of the spiral search algorithm. It functions in much the same way except that pattern is that of a box. This allows the pathing to remain strictly linear or rotational and has the added bonus of matching the shape of the arena which limits the rover's ability to trigger a collision command with the walls.

The out and back algorithm has the rovers picking a random compass heading and going in a straight line until the arena wall was reached and then turning around and returning to the center. If a resource was detected along the path the rover would repeat that path until no new tags were discovered and then pick a new random heading to search. This search method has the benefit of being very simple but has an efficiency drawback. To sufficiently cover the far reaches of the arena, the paths have a lot of overlap closer in to the center of the workspace. So, this causes a tradeoff of minimizing overlap in the center of the arena at the cost of missing areas further out or taking more time to search to cover the arena thoroughly. Since the distribution of tags is random and unknown ahead of time, there is no way to make an educated guess on how to best distribute the search lines.

The rake search algorithm has the rover moving across the arena in the 4 cardinal directions navigating mainly with the compass and ultrasound sensors. The algorithm can be broken down into four main steps: runway, position, sweep, and return. This algorithm has the rovers all moving across the collection zone in one direction (runway) and then choosing a y-coordinate to search (position) once the west wall of the arena is found. The algorithm also allows the rovers to reset their location data should they get lost by searching for two walls signaling that it is on one of the four corners of the arena.

Figure 7: Scale testing of the algorithms on the physical platforms

The search methods that made it out of simulation described above were all put through physical testing and iterated on. The physical rovers did not act in the idealized manor that they did in simulation often causing results that varied from the simulation, mostly the rover's ability to accurately track its location was prone to error leading to issues finding the collection zone once a tag had been picked up. The codes were either modified to try and account for the results of the physical testing or scraped if a workaround was unable to be found.

Final Design and Results

After testing and reworking the existing codes, the result was the rake search algorithm being the most reliable method, although not the most efficient as it was required to traverse a lot of distance per search loop. Yet a more efficient algorithm is useless if it quickly becomes unable to keep track of its location. Over the course of testing the rovers the main issue that arose was a large amount of drift that would build up in the rover's wheel encoders. After only running for a minute, the rover could think the origin point is over a meter away from where it started. The problem, if left unchecked, would just allow for this error to continue to build, quickly causing issues with returning to the nest to drop off collected tags.

The solution to this issue is to heavily rely on the data from the rover. The algorithm relies heavily on the compass and ultrasound data. Through testing, they were the most accurate and reliable. The four walls and the nest tags were used as landmarks that allow the rovers to switch between one of the six modes previously programmed. There are four main modes that allow the rovers to navigate the field and collect and return tags, an initialization mode, and a recalibration mode that allows the rover to reorient itself within the arena if it is unable to find the nest or encounters and unexpected collision.

The initialization mode (yellow) simply moves the rovers from the center into a position where it can start the main tag seeking modes. This initialization also lets the rovers know what round they are in so they can search the proper amount of area. The four main modes can be broken down as runway (green), position (blue), sweep (orange), and return (red). The runway mode is an east to west path along the x-axis that passes through the nest. The idea for this is like a runway at an airport. The team have the rovers passing though the nest all in the same direction such that they will limit the possibility of coming into close proximity with one another. The position mode assigns a y-coordinate value to the rover and has the rover either moving north or south to get to the desired y-coordinate. The sweep mode is a west to east path that has the rover looking for

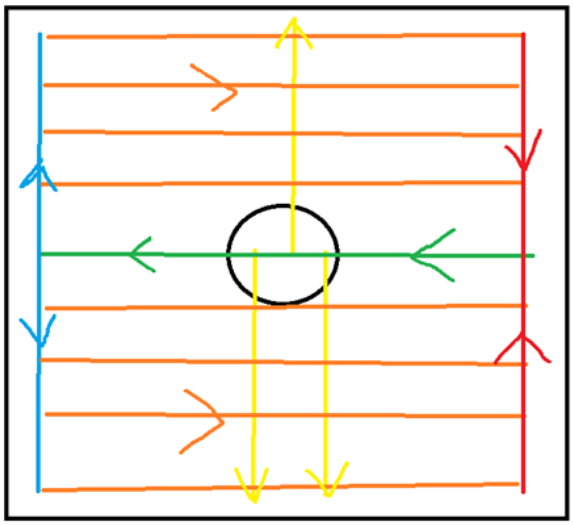

Figure 8: Diagram of the rake search algorithm

tags as it moves back across the arena. The return mode has the rover moving north or south to return to the x-axis so that it can go back to the runway mode.

At the end of the runway mode the program performs a check to see if it has passed by the nest. If it has it continues as planned, but if it missed the nest that means the rover is now where the program thinks it is and it sends it to the recalibration mode. The recalibration mode has the rover, depending on that side of the arena it is on, go to the north or south wall then go to the east wall. By using the ultrasound sensors and the known arena configuration we are able to move the rover into an easily found known position (either the northeast or southeast corner of the arena). With the rover back in a known location, the rover is returned back to its normal operation.

The algorithm takes a simple principle of having the rovers rake across the arena in search of tags and returning them in a way that limits the possibility of the rovers approaching each other head on or in any other way that would interfere with their pathing. This also allows for the rovers to work around the fact that their positional data accumulated large amounts of error. As the only point where the algorithm directly references the positional data is when having it position in the y-direction. But instead of assuming the y-value is constant, we always take the current location value and have it move north or south a relative amount corresponding the desired y-coordinate. This limits the error to a single movement that is at most seven meters in the preliminary round, and should there be an error that is significant enough to have the rover be in a position that is appreciably far away from where the algorithm believes it to be we have the recalibration step. There are algorithms that could find and return the tags in a much faster manner if the

positional data was accurate. But this is an algorithm that should work reliably and be able to overcome any tracking errors assuming the compass and sonar data is reliable.

During the competition, and unforeseen complication greatly hampered the ability of the rovers to function properly leading to them only being able to collect a few tags. Since the rovers are mainly guided by the compass and ultrasound I wanted them to be as precise as possible when turning to their headings. This lead to having as tight of a tolerance possible for the sensor, about 4 degrees, and having the code update that value within the loop of having it perform its linear motion and revert back to the rotational portion should the heading deviate too far. We had followed the instructions to properly calibrate the gyroscopic systems that included the compass and had no issues while performing full scale testing of this method. However, at the competition there was either noise, interference, or a lack of calibration that cause the compass value to deviate out of the acceptable range frequently enough to cause the rovers to mostly stutter and keep attempting to correct their heading instead of performing the desired linear motion. This wasted a lot of time with the rovers sometimes having a moment of clarity to perform the desired pathing. The main way to alleviate this issue was to remove the ability to update in the linear motion part of the code so it would be able to move undisrupted, however since the rovers had to traverse 15 to 22 meters in as straight a line as possible, that is why the value was constantly checked so the pathing would remain straight if the rover starting veering.

2017 Competition

Changes vs 2016 Competition

For the 2017 competition, 2-DOF manipulators were added to the rovers and they now picked up physical cubes. While the overall approach can be the same, there are some factors that now need to be considered. The path planning needs to be conscious about not going through the center as the rovers can push cubes out of the collection zone. The rovers also need to try and return to the center via already cleared areas or else cubes could be moved around and possibly into the areas that the rovers have marked as previously cleared.

Swarm Algorithm Development

Having the knowledge of the previous year, the algorithm to run the rovers needs to be accurate enough to find a resource and return to the center but also flexible enough to account for the inaccuracy and variance between the rovers, especially with regards to their positional tracking systems. Before a new search algorithm was worked on, ways to improve the way the rovers communicate and how to better use the sensors in a more accurate manner such that we did not have to rely so heavily on the compass and ultrasound as we did last year. As far as communication goes, the rovers could broadcast their location when they find resources to alert the others to come and help, but based on previous testing, the rovers work better when spread out as if they are all attempting to go to the same spot they trigger each other's ultrasound sensors causing them to all detour around each other perpetually. The location of the rover is still useful since it can be used to have the rovers avoid each other minimizing time wasted through obstacle calls if they pathed at each other.

This year the Arduino software was updated to allow for them to make a spiral motion so that code was tested again on the physical platforms. The rovers were able to make the spiral motion, however due to having 4 fixed wheels, it would occasionally slip or overturn depending on the ability to grip the ground. The other issue of finding the center still persisted as the location being tracked had larger errors the longer the rover pathed.

Since the positional errors stacked as successive moves were made, a chessboard search algorithm was developed to be flexible enough to handle the error but accurate enough to have the rover consistently find the center. This method breaks down the arena into 1 meter by 1 meter squares and treats them like spaces on a chessboard. A goal is assigned as a space number and the rovers move to that space by making steps from one space o the next until they arrive at the desired space. With a space this big, we can be very confident that the rover is within the space we expect it to be in. And since the collection zone is also a 1 meter by 1 meter space, when the rover moves into that space, we can be confident that it makes it there. If the rover doesn't read the tags when it expects to be in the center it performs a recalibration step that had been lifted from last year's code.

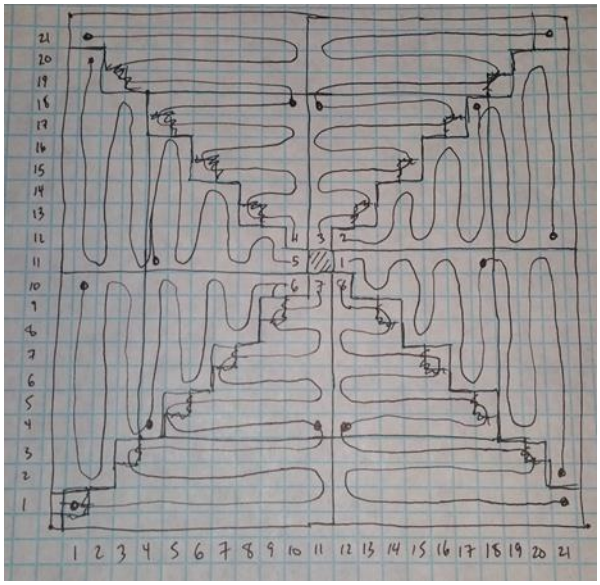

Figure 9: Sketch of chessboard search algorithm with pathing for each zone capable of navigating both arena sizes

Final Design

For our ultimate design, we decided to break down the arena into smaller discrete spaces that are 1 meter by 1 meter squares. This allows us to cover

the small arena with a 15x15 grid and the large arena with a 21x21 grid. Using the compass and 1 meter straight pulses from the rovers, we are confident that the rover will be somewhere within the space we designate it to be at.

Each time a rover arrives at a space it has never been to before, it performs a 360 degree sweep of that space to check for resources in the vicinity. If any are found it retrieves one and makes its way back to the collection zone backtracking the most direct route using previously cleared spaces. This reduces the chance of the rover accidently pushing resources into areas of the map that it has already cleared. Upon dropping off a resource the rover can confirm is location using the collection cubes as a landmark and then return to the space that it collected that previous resource from. It will again perform the 360 sweep to make sure all resources have been cleared from that space before moving onto the next space and repeating the process over.

The arena has been broken down into 8 zones, each one having an origin space on of the 8 squares that are adjacent to the collection zone. Having 8 zones allows for rovers to be in different zones and all have a clear access point to the collection zone. Each zone has a fixed path that snakes from the center out to the perimeter of the arena. Upon arriving and clearing the final space of a zone, the rover will move to the next zone over and reset its position at the new zones origin space.

Figure 10: Simulation result for a zone's pathing with the little circles representing the 360 sweep at each new space

In addition to the above, there are built in measures if the collection zone is

not reached or a wall is encountered in an unexpected location. In both situations that means the rover is not where the program thinks it is. In this case, we trigger an automated recalibration which has the rover going to one wall, then to an orthogonal wall of the first. This allows us to reset the x- and y- coordinates using known geometry of the arena. From there it can proceed back to the center or to its previous space it was searching.

The chessboard search method allowed for us to relax the tolerances on positioning to a level that we felt could be reliably maintained and the rovers are accurate enough to make 1 meter cuts in a given direction. This method allows for us to track the rovers in terms of a 1 meter by 1 meter space that they occupy instead of a coordinate that could accumulate a tracking error over the course of the run.

Through testing in both simulation and on the physical rovers, we feel confident that the rovers can detect, pick up, and return the cubes in a quick and efficient manner with minimal need to have a positional recalibration. The one concern is a scenario when multiple rovers return at the same time and detect each other. However, that case should be infrequent enough that if it were to happen and they were sent to recalibrate it should minimally impact the performance.

Proposed Improvements

The proposed vehicle and controller scheme are for a system of automated trackless vehicles working together in an industrial setting.

Vehicle Design

The proposed vehicle takes cues from various vehicles to come together and form a highly maneuverable platform that is capable of operating autonomously in an industrial

environment while performing desired tasks and carrying payloads. The dimensions of the vehicle can be scaled to meet the needs of individual work environments.

The main feature of this platform is having four electric drive motors that drive each of the four wheels. Each of the wheels are also capable of being rotated in a full 360 degrees of motion by an additional four motors. Having all four wheels fully powered and controlled, as opposed to current platforms that have two drive wheels and two castor wheels, allows the wheelbase of the vehicle to be expanded for larger applications in addition to giving increased power and traction capability.

The main benefit to having 4 independently powered and steered wheels allows for the vehicle to move in any direction with any orientation. This gives much more control over traditional rack and pinion steering or even vehicles capable of crab steering. This is very useful in situation that either need to have vehicles moving into very precise positions since they can adjust in any direction without having to pull out and back in like a traditional vehicle would. This can also allow them to navigate much tighter locations since they can rotate on the spot and change direction while keeping their orientation.

The vehicle will be equipped with a wide array of sensors for monitoring its location and surroundings.

Control System

The proposed control system is to use and build upon elements from previous control systems such that efficiency and reliability are increased.

A hybrid controller will be employed for the system having a main controller monitors all sensors, both environmental and those on each vehicle, and control the fleet of vehicles on a macro level with path planning and task queueing. Each vehicle will also

be equipped with a controller able to read from its sensors and perform on board calculations and take independent control of the vehicle when either contact with the main controller is temporarily lost or for a reactionary maneuver that would take too long for the central controller to sense and react to.

The motivation for having multiple control layers for this system is to increase reliability of the system. With these vehicles being trackless, they are not hardwired into the control system and often rely on a Wi-Fi or other close ranged network signal that can fluctuate in coverage or suffer from slowdown when too much information is being transmitted. If control was handled purely from a central system, any network interference would cause an affected vehicle to stop and wait for either the signal to return or for a manual override.

The system will have a wide array of sensors, both onboard the vehicles and in the environment.

The primary positioning sensor will be the 3D spherical emitter and sensor arrays. Each vehicle will have of these 3D sensors, on at each corner of the vehicle. This will allow for the sensors in the environment to be able to always have a clear view of at least one of the sensors regardless of vehicle orientation and also allow for it to calculate the orientation of the vehicle. These sensors not only allow for the central system to read the position of the vehicles, but also allow the vehicles to read their position in the environment and allow them to read other vehicles that they have line of sight with.

The benefit of each 3D sensor is that it both emits and reads so that the information is always being shared both ways between the various systems. This allows both layers of

the control system to independently track position and verify the data between them and also allow for them to operate independently should one end fail or lose contact.

A GPS sensor could be used for each of the vehicles if the application has a large outdoor workspace where it would be uneconomical or impractical to cover the whole workspace with positional sensors.

The vehicle themselves are equipped with an accelerometer, load scale, positional and velocity tracking for each of the vehicles wheels, proximity sensors, and cameras.

The goal of the main system level controller is to be the manager of the entire system. It should be able to receive inputs from outside the system, such as user input for desired tasks, and then assign that to an active vehicle in the fleet. In assigning the tasks, the system needs to take into account the positions of all the vehicles and their task status. It should assign new tasks as they come in such that it is carried out in the most expedited manner without affecting the flow of the rest of the tasks. The main system also calculates the path that the vehicle needs to take to carry out this task. The path planning is fully calculated before the task is assigned so that after the vehicle is assigned the task from the system, it could complete the task even if its connection with the main system is interrupted. While the task is being completed the two systems share all their sensor data to cross reference for verification and to monitor the progress of the vehicle.

The vehicle level controller is a simpler system that just takes the assigned pathing data and convert that to the required outputs for each of the four wheels. The vehicle controller also monitors onboard sensors that are mostly to monitor the surroundings for possible obstacles that would need to be avoided.

With four independently powered and rotatable wheels there are a lot of factors that go into the outputs for each of the wheels. Since they are not physically synced and are capable of being operated at different orientations, the precision of the controller is paramount since loss of wheel control when the vehicle velocity is sufficiently perpendicular to the wheels can cause damage or abrupt stops that could cause the load to be jolted loose. The two data points that are fed into the vehicles system are the desired Cartesian position of the vehicle along with its desired orientation. Each of these can be calculated as separate components of the wheel outputs then combined afterwards to get the independent signals for each wheel. This simplifies the calculations since the component of the wheel output for positional data is the same for each of the wheels. The orientation component of the wheel output is in a direction tangent to a circle that is centered on the middle of the vehicle and passes through each of the wheel's locations. This output is affected by the dimensions of the vehicle so it is very important that those are represented correctly in the system. Each of these components of the wheel output are linearly combined to give the required independent outputs for the wheel velocities and headings.

Positional Data

Having the vehicles use 3D relative positioning sensors they do not need to rely on more conventional methods such as GPS or wheel encoders, which each have shortcomings, lack of precision in some environments and error accumulation respectively. The only downside to using the 3D relative positioning sensors (3DRPS) is that you have to make sure that there are enough environmental sensors within line of sight of the vehicle in its workspace. With the vehicles and environmental sensors having a broadcasting and

reading sensors, each can determine the location independently and cross reference for accuracy.

Each vehicle would be equipped with a minimum of two sensors in known locations on the vehicle. The environmental sensors are located in known locations within the vehicle workspace.

For the master system to determine the location of a vehicle it requires that one environmental sensor reads two sensors on the vehicle to resolve the 2 unknowns of location and orientation. If multiple environmental sensors can read the vehicle, then multiple reading can be cross referenced to improve reliability and reduce the possibility of a rogue reading or noise.

The vehicles have two possible methods for determining its location and orientation. The first way to determine its location is by reading its distance from a known location of an environmental sensor plus a compass reading (heading). The second method is to have one of the vehicle sensors read the locations of two environmental sensors.

The above methods for determining the vehicles location are the minimum data points required and more can be used for verification purposes.

Vehicle Controller

The vehicle has a total of eight motors that work together to provide the desired vehicle movement. There are four drive motors (DM), one for each of the drive wheels which are controlled by setting a desired velocity. There are four rotational motors (RM), one for each of the drive wheels that control the angular orientation of each wheel indecently. They are controlled by giving a desired angle.

The vehicle will only need three inputs for the controller to sufficient define the required output. The inputs will be the desired x- y- coordinates of the location and the phi angle, which is the orientation of the platform, not the heading of the vehicle motion. The theta is resolved within the controller by solving the trigonometric relationship between the current vehicle location and the desired vehicle location.

The controller will use the desired values for the inputs, supplied from the main control system, and use a PID controller to determine how each will factor into the two output values for the vehicle motors. Each value will use feedback from sensors for each of the respective inputs to allow for maximum accuracy and responsiveness. Once the framework for the controller is built, the gain values for each input will be optimized to give the best balance between a smooth, yet responsive signal to the output motors. Since the controller can be adapted to vehicle size and environmental needs, the gains for the controller will need to be fine tunes for each application independently.

The velocity and angle for each of the four drive wheels are the result of the desired liner velocity and heading along with the desired platform orientation. Each of these events can be calculated separately then combined linearly to determine the final outputs for each of the wheels.

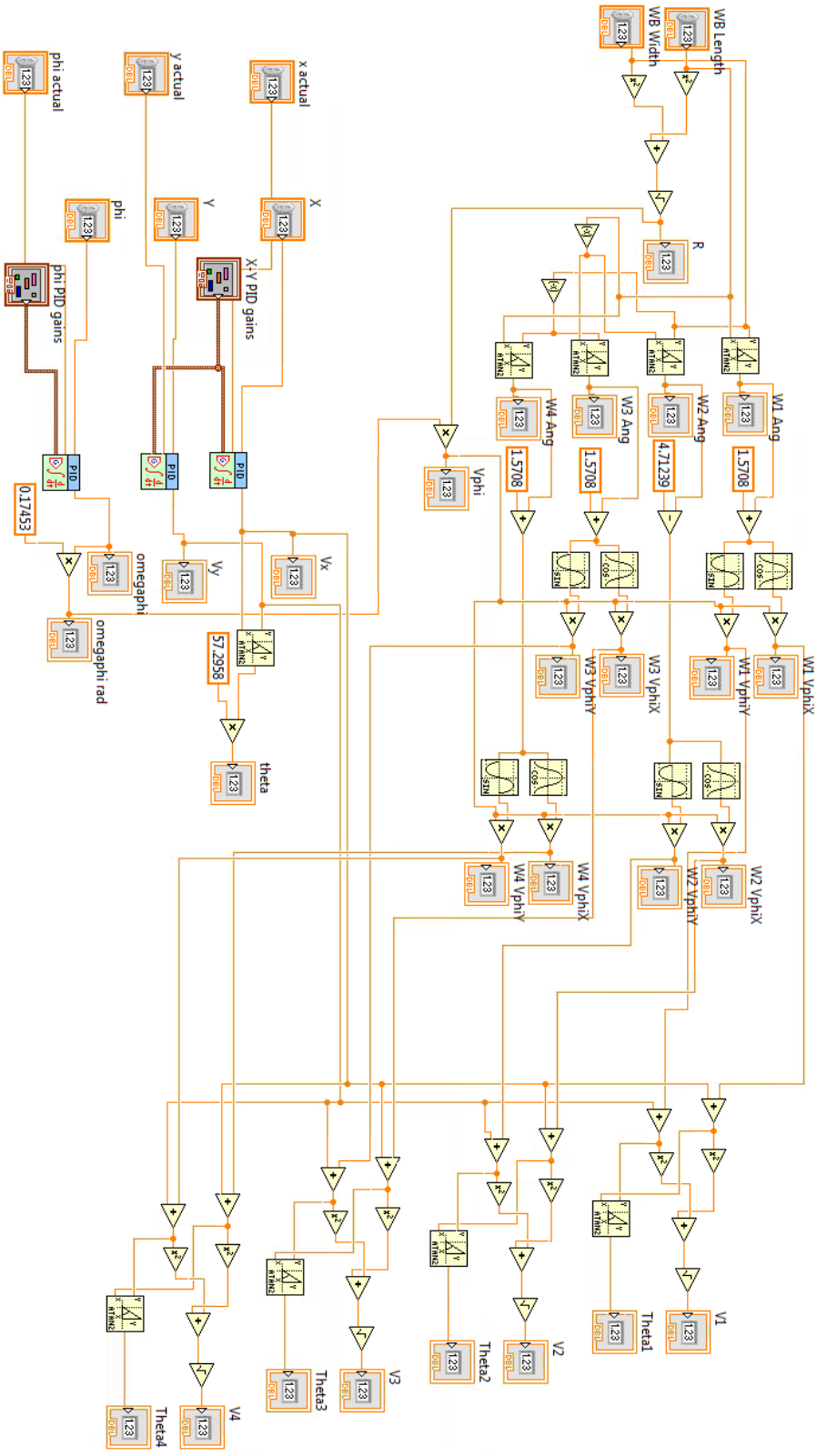

Figure 11: PID controller for vehicle motion

Final Design

Vehicle

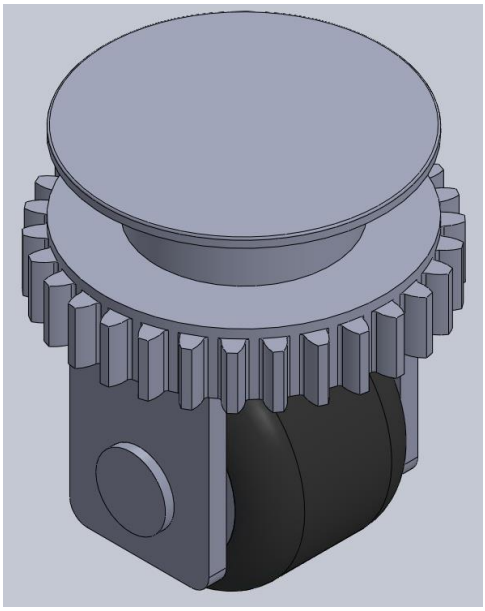

Figure 12: Wheel with mounting bracket.

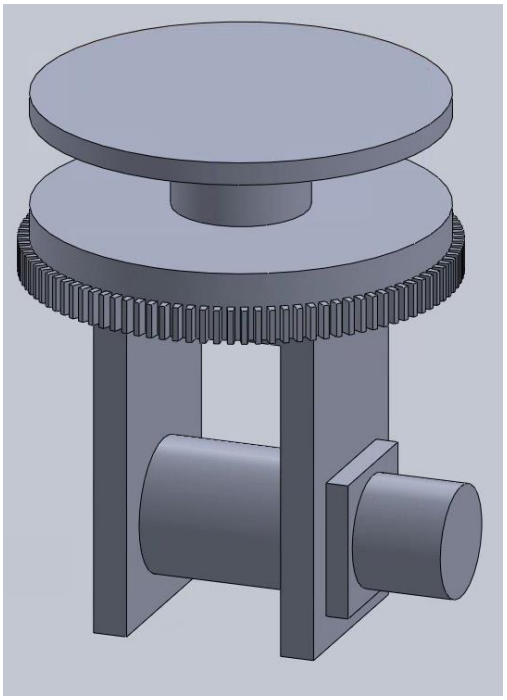

Figure 13: Wheel bracket showing where the drive motor is mounted onto the side of the bracket.

The design of the vehicle allows for size scalability and maneuverability for a wide array of uses. The vehicle has a very simple design consisting of 3 main parts.

First is the wheel and mounting bracket. This consists of the wheel, the bracket that the wheel is mounted to, and the drive motor for the wheel. Since the wheels are meant to spin a full 360 degrees the bracket must be able to easily rotate with respect to the main platform while also being able to stably support the loading of the vehicle. To account for both these needs, the cuff of the bracket has a wide plate on the top and bottom that help distribute the loading for each wheel assembly while also limiting the frictional forces that need to be overcome for rotation of the cuff to be performed. Figure 7 shows an earlier design of the wheel and cuff. As can be seen midway up the bracket, there is a gear built into it. This is for the second motor to interface with to control the wheel orientation. In figure 8 the placement of the

drive motor is shown on the outside of the bracket lined up with the axis of the wheel. The bracket can either have one motor mounted to one of the sides, or two motors working in tandem mounted on both sides to amplify the power available to the wheels while keeping the size of the motors in check.

Second is the lower platform. This is the main housing for the vehicle as it is where the wheel brackets link into as well as housing the required mechanics for the vehicle such as power source (batteries or generator), the motors that control wheel orientation, most of the onboard sensors, and the computer system.

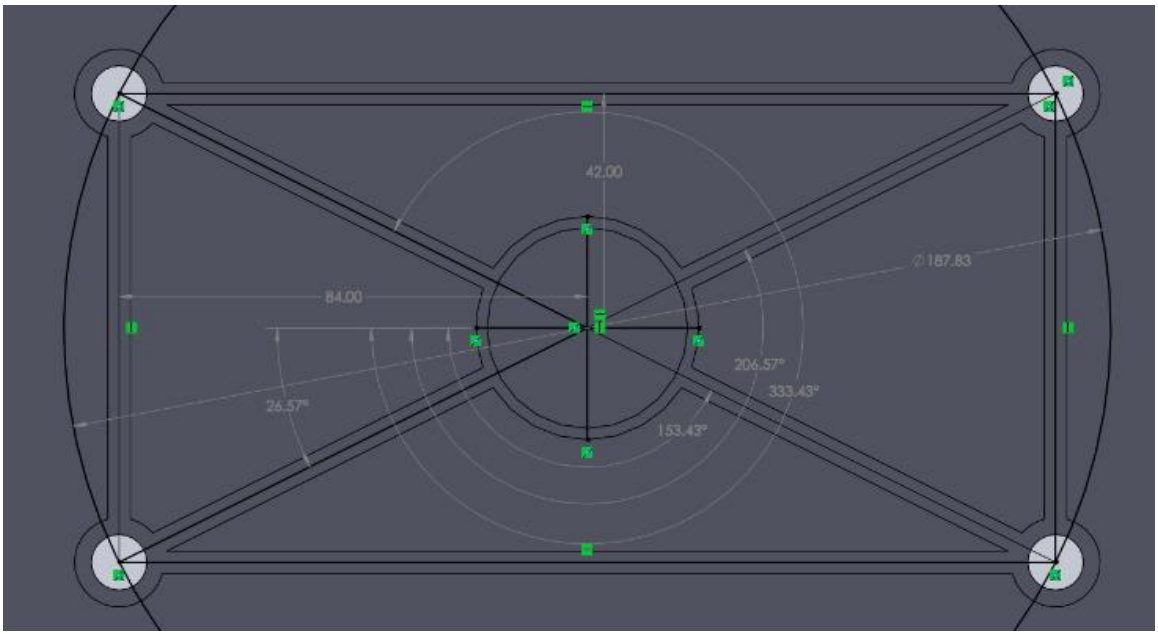

Figure 14: Underside of platform showing a possible wheel configuration with respect to the center of the vehicle.

To make control of the wheels as simple as possible, they are laid out along a circular pattern with respect to the center of the vehicle. By having the wheels laid out in a circular pattern about the center of the vehicle, it creates an easy reference not only between the central point of the vehicle, where the position will be taken from, but also an easy reference between wheels so that rotational maneuvers have the most evenly distributed

output requirements. This layout allows for the required wheel velocities and orientations to be easily calculated getting eight outputs (velocity and orientation of each wheel) from three desired inputs (liner velocity, direction, and vehicle orientation). Knowing only the length and width of the wheelbase, the location of all four wheels can be resolved for the system to properly calculate the required motor outputs to the wheels.

Since each wheel is independently controller that mean each wheel has two motors, one each for linear and rotational velocity outputs. Each wheel system is connected to the main platform via a cuff that is designed to be able to carry the required loadings for the vehicles desired application. The rotational motor is mounted within the platform and interfaces with the wheel system via a gear. The tricky part comes with the mounting location of the drive motor. There are two options, on the platform or directly to the wheel mounting bracket. Mounting from the platform would require a driveshaft that can not only operate at an angle, but be able to compensate for the rotational movements with the vehicle without affecting wheel velocity. Mounting from the wheel bracket gives an easy, direct connection to the wheel but creates an issue with wiring back to the platform to be able to receive power and a signal. Mounting to the wheel bracket was determined to be preferable since slide contacts can be used for the transfer of power and required data.

Figure 10 shows how given a desired linear (red arrow) and rotational (blue arrow) velocity for the vehicle as a whole, it can be broken down to the wheel components which combine for the total output for each wheel (green arrow). Notice how the blue arrows for each wheel are always tangent to the circle that the wheels all lie on. The resultant motion shown in figure 10 is of the vehicle driving in a straight line while its orientation is rotation counter-clockwise.

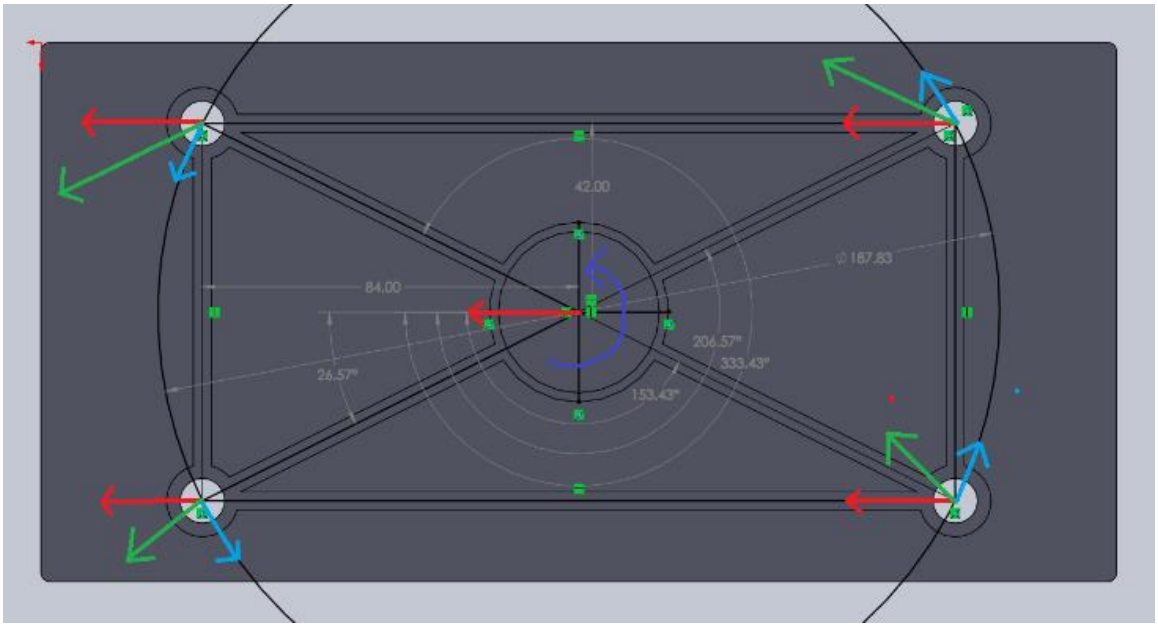

Figure 15: Desired linear and rotational velocity depicted in the center result in the wheel outputs (green, with red (linear) and blue (rotational) components

Figure 11 shows a design for the lower platform where there are large recessed areas for the wheel bracket to slot into along with ribbing along the platform to reinforce it in areas where forces are expected to be transferred along the body of the vehicle. The positional sensors can also be seen mounted at each of the corners of the vehicle. Having a sensor on each corner allows for an environmental sensor to be able to see at least two of the vehicle sensors at a time assuming no other obstacles possibly blocking the line of sight besides the cargo onboard. Figure 12 shows the same lower platform with the wheels in place. The top of the bracket cuff is flush with the platform ribbing and the built-in gear are exposed below the platform. This allows for the orientation motors to be mounted to the lower platform and have them connect to the wheel brackets via a hole in the floor of the platform.

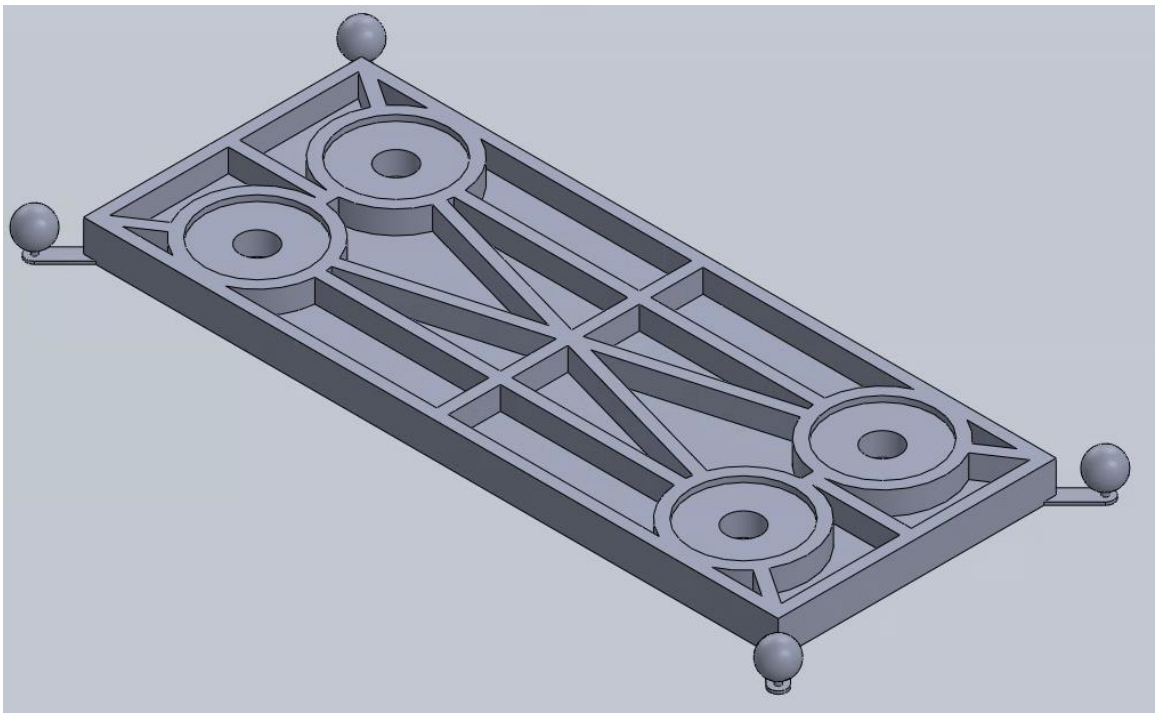

Figure 16: Lower platform complete with cutouts for wheel brackets and positional sensors mounted to corners.

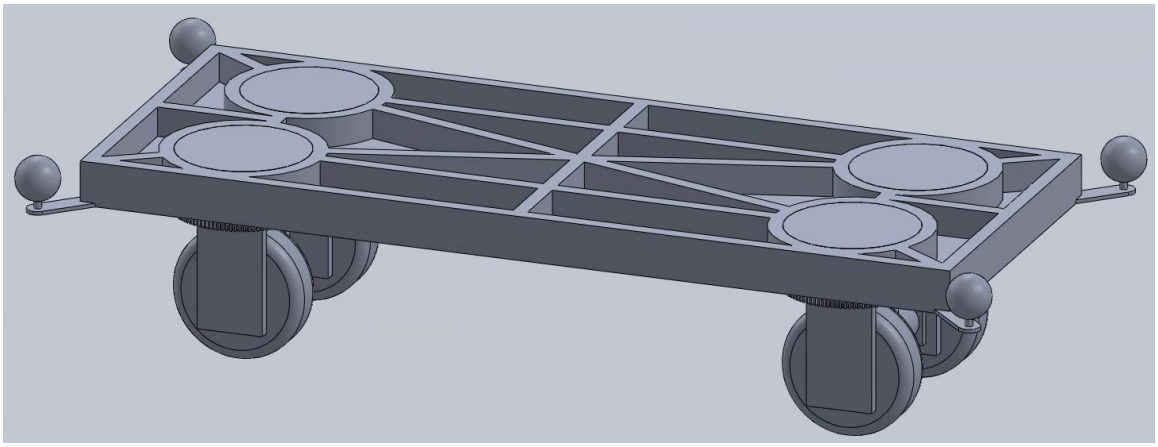

Figure 17: Lower platform with wheels in place.

Third is the upper platform. The upper platform sits on top the lower platform covering and protecting the mechanical components of the vehicle while also providing the required base to interface with the desired cargo be it pallets, shipping containers, or people. The upper platform can be designed with a specific requirement in mind. Figure 13

shows an upper platform that is designed to hold a 20x8 foot cargo container. The modular design allows for the upper platforms to be interchangeable with the main vehicle allowing for one vehicle to be used with many types of cargo without having to compromise on how the cargo interfaces with the vehicle.

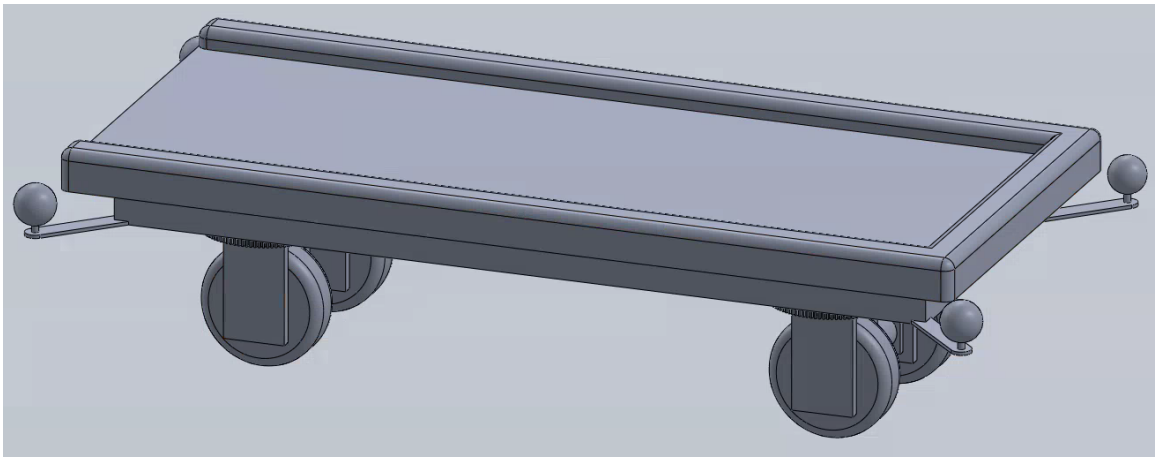

Figure 18: Compete vehicle with upper platform that is designed to handle large shipping containers.

of a main platform that serves as a housing for all mechanical parts upon which the cargo load can be placed. With the goal of having a highly maneuverable platform, it is essential to have the wheels laid out in a configuration that gives proper stability to the vehicle but also makes it easy for the control system to control the vehicle for more advanced maneuvers. By having the wheels laid out in a circular pattern about the center of the vehicle, it creates an easy reference not only between the central point of the vehicle, where the position will be taken from, but also an easy reference between wheels so that rotational maneuvers have the most evenly distributed output requirements. This layout allows for the required wheel velocities and orientations to be easily calculated getting eight outputs from three desired inputs (liner velocity, direction, and vehicle orientation). Knowing only

the length and width of the wheelbase, the location of all four wheels can be resolved for the system to properly calculate the required motor outputs to the wheels.

Since each wheel is independently controller that mean each wheel has two motors, one each for linear and rotational velocity outputs. Each wheel system is connected to the main platform via a cuff that is designed to be able to carry the required loadings for the vehicles desired application. The rotational motor is mounted within the platform and interfaces with the wheel system via a gear. The tricky part comes with the mounting location of the drive motor. There are two options, on the platform or directly to the wheel mounting bracket. Mounting from the platform would require a driveshaft that can not only operate at an angle, but be able to compensate for the rotational movements with the vehicle without affecting wheel velocity. Mounting from the wheel bracket gives an easy, direct connection to the wheel but creates an issue with wiring back to the platform to be able to receive power and a signal. Mounting to the wheel bracket was determined to be preferable since slide contacts can be used for the transfer of power and required data.

Control

The system for the trackless vehicles is a two-tiered controller that works in parallel. This consists of a master controller that oversees the entire system, and a local controller that is on board of each vehicle that carries out the plans of the master controller.

The higher-level controller is a system monitor and planner for the entire system. There are multiple facets to this controller.

First, this controller plans tasks and distributes them down to the individual vehicles. This is done by the controller taking a queued task in the system and deciding which vehicle will complete the task. This can be based on proximity or which vehicle was

used last or many other parameters. The controller then creates a path for the vehicle from its current location to that where the task needs to be completed. The path is given in terms of x- and y-coordinates and the orientation of the vehicle (phi). These are all given with respect to time to control the velocity of the vehicle depending on loads such that it can safely stop given the range of its onboard sensors.

Second, the controller monitors all the sensors that are hardwired into its system. These are all the environmental sensors so that the controller can make informed decisions on path planning and current work flow. This can allow the system to read obstructions so that it can plan ahead of time instead of having the vehicle itself sense the obstruction and have to react locally.

Third, the controller receives all the sensor data from each of the vehicles and verifies that data with its own sensor data. Since the positional sensors have a component on both the vehicle and in the environment, the reading from each of the perspectives can be compared since they should ideally be reading the same value. The difference in value is compared to an acceptable tolerance that can trigger a shutdown if the error exceeds it. Since the controller can read the vehicle sensor data, it can make updates to the path should the vehicle encounter a local obstruction that the overall system did not account for when originally planning the path. This allows for the system to adapt in real time using data points from any of the vehicles. While the vehicles do not have to directly connect, the system can use data it receives from one vehicle and apply that information to any other vehicle that the system deems would benefit from the data.

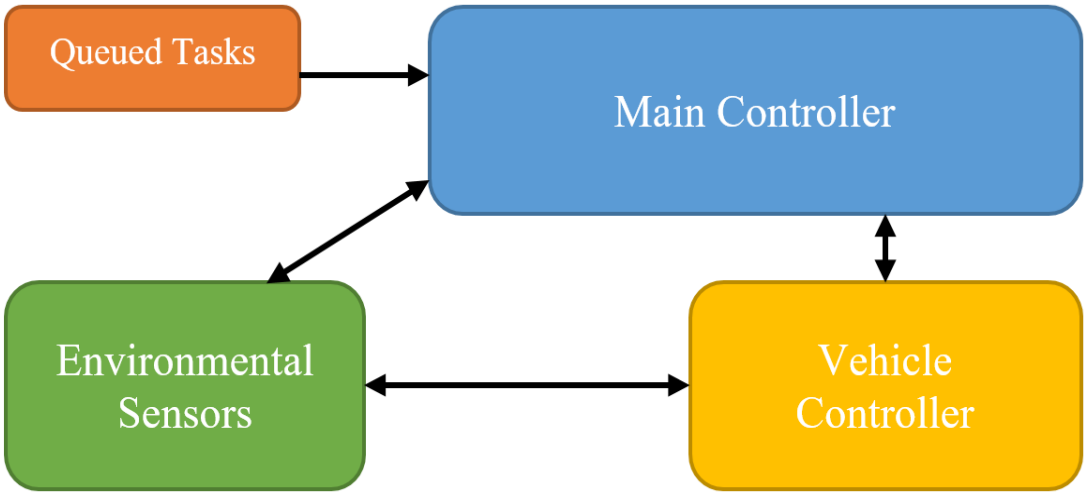

Figure 19: Flowchart for flow between controllers and sensors

The vehicle controller is fully capable for operating a vehicle independently of the main controller. The only pertinent information the vehicle gets from the main controller is the path to a goal location. Having the vehicle contain a full suite of on-board sensors and processing capabilities allows for the vehicle to navigate the work environment once the path data is supplied to it. This allows for the vehicle to keep running even if the connection is lost with the main system for whatever reason. Once the vehicle reaches its desired goal position, if a connection is still unavailable from the main system, the vehicle will wait. While a major connection issue will cause productivity to stop in some capacity depending on the location of the

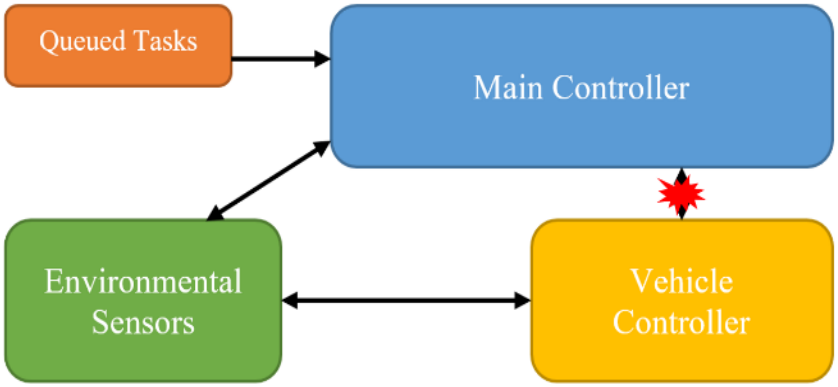

Figure 20: System with connection issues between main controller and vehicle

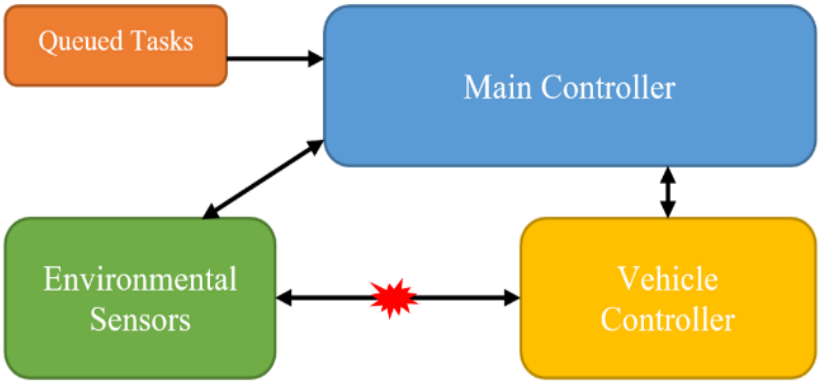

Figure 21: A signal error between the vehicle and environmental sensors

connection issue, the main benefit on this is that intermittent connection issues will not force emergency stops or lead to vehicles losing control if they relied on the main signal for motor output signals. This can maintain a high uptime barring a major issue with the main controller. With most of the work offloaded and distributed to the vehicles, a vehicle controller issue is minimized to only the affected vehicle and will not impact the performance of any other vehicle in the workspace.

Besides a connection issues between main controller and vehicle, there are other issues that can be overcome by the layout of the system. The positional sensors that are employed for this system are 3D relative positioning sensors (3DRPS). The way these sensors work is that each sensor both emits a signal and reads the signals from other sensors. The 3DRPS are placed in known locations of the environment. This allows the environmental sensors to read the relative position of a vehicle and report that to the main system. This also allows the vehicle sensors to read the relative position of the environmental sensors and calculate their own location. Since the vehicle is having its position read from two sources, this creates a level of redundancy that can be used for verification purposes to increase accuracy, but also to allow for operation to continue if one of the signals is either unable to read or giving bad numbers. This can mainly be used to allow for the system to identify and ignore possible rogue values from the sensors

without having the system have to stop. While this can be used to allow for operation to continue if there is a more permanent issue with part of the sensor network, it would be recommended that the vehicle only operate to finish its current task then remove itself from service to be inspected. However, even if a vehicle does have some kind of sensor error, the redundancy and parallel structure of the system allows for the vehicle to finish its task and exit the workspace versus having to perform an emergency stop while loaded in the middle of its work environment.

The main part of the vehicles controller is a PID controller that takes the desired path generated by the main controller, in terms of x- and y-coordinates and a phi angle (vehicles orientation) and calculates the required vehicle outputs in terms of wheel velocities and headings. The controller's feedback uses the sensor data from the 3DPRS to compare the vehicles current location with the desired location described by the intended pathing. The vehicle also contains other on-board sensors that are for local obstacle detection. There are three ways the vehicle controller uses the local obstacle data. If there are no detections or an obstacle is at a distance that is out of the vehicles path envelope, the vehicle will proceed uninterrupted. If the vehicle detects an obstacle that is along the planned path of the vehicle the controller can modify its path to avoid the object. If the vehicle detects an object within a proximity that the vehicle deems is nearing its stopping distance, it will tell the vehicle to stop. There will be a sufficient cushion to the stopping distance that the

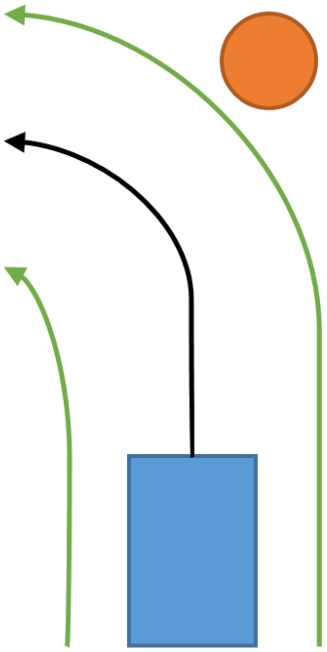

Figure 22: Vehicle detects obstacle but it is out of path.

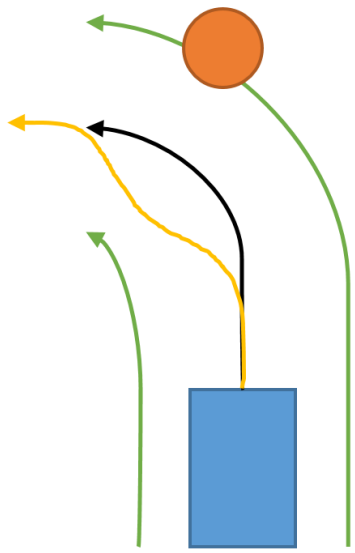

Figure 23: Vehicle detects obstacle in path and makes course adjustment.

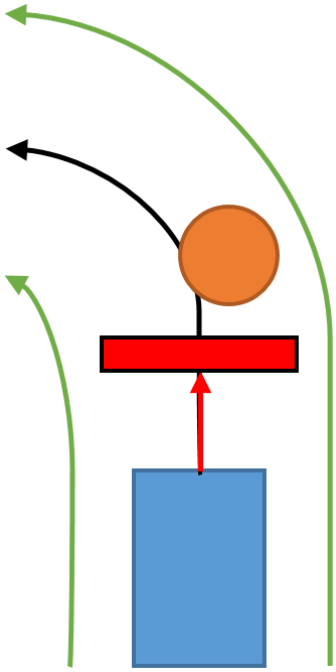

Figure 24: Vehicle detects close obstacle and makes a full stop.

vehicle will err on the side of safety. The third condition of having the vehicle stop is only a result of extreme circumstances such as obstacles falling into the vehicle path or a moving obstacle that happens to be moving in the same direction the vehicle is attempting to deviate its path to.

The features of the main controller and vehicle controller come together to integrate with a multitude of sensors that allow for improved safety and efficiency over current applications of automated vehicles. This system can be deployed for virtually any application so long as there is enough infrastructure to establish a sufficient network of environmental landmark sensors. The main controller can be calibrated and scaled for any size and application from transportation of shipping crates in a shipping yard, moving raw material in a factory, or controlling a fleet of ride vehicles in an entertainment application.

Discussion

Improvements Developed

This research has resulted in an improved design and control for an autonomous guided vehicle that allows for a more forgiving control system that is less susceptible to downtime from intermittent connection between the main system and vehicles. This benefits industrial environments that have a lot of signals and other possible sources of interference that could lead to an unstable wireless connection between system and vehicle. Besides a more distributed and redundant controller, the vehicle proposed is more capable in both power and handling by increasing the number of powered and fully rotatable wheels from two to four over what currently exists in industry today.

From working with the swarm rover, if not appropriately accounted for in the controller logic, having an interruption between vehicle and controller can cause the system to stop, pause until reconnected, or in the case of the rovers, loop on the command being performed when the connection is lost. For small rovers working in a controlled environment that is not a safety risk should the connection be lost and the rover keep going straight indefinitely. However, for a vehicle that is working around or carrying people, having the vehicle continue moving in an uncontrolled manner cannot be allowed. While the control method proposed has the vehicle capable maneuvering with a severed connection, it does so only if it is still able to read all sensors that the vehicle is equipped with to safely make maneuvers. Should this extend past a predetermined amount of time or until the currently designated task is complete, the vehicle could then wait for connection or safely move itself out of the workspace and await repair if the connection issue is on the vehicles end.

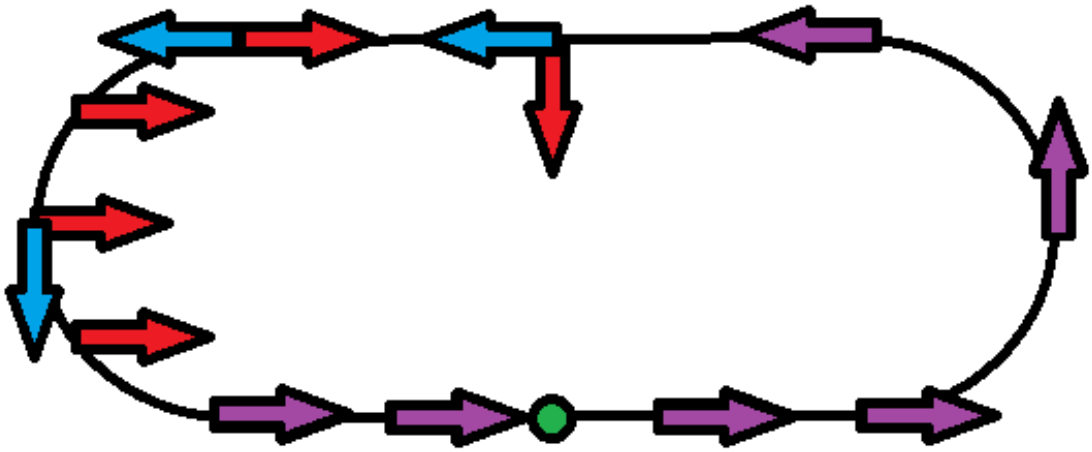

Figure 25: Desired path and orientation of the vehicle.

In Appendix 1 there are the results for the positional controller for a vehicle. It follows. The vehicle starts at the green dot and follows the path counter-clockwise. The arrows denote the direction of liner motion (blue arrows) and the orientation of the vehicle (red arrows) or in the case when the two are the same by a purple arrow. The path has a horizontal scale of 300 meters and vertical scale of 100 meters. Following the path at a constant 5 meters per second, one lap is approximately 143 seconds. The results show that the vehicle is able to follow the path rather well with minimal errors and fluctuations in values. The maximum errors were 0.57 meters for the xy-coordinates, 2.36 degrees for the direction heading and 3.84 degrees for the vehicle orientation. The coordinate error starts off low and builds up to a point where the x-component is off by a steady 0.5 meters after it comes out of the final turn. The direction heading fluctuated minimally to keep the vehicle on path and the orientation only had errors while it lagged behind in the desired rotation, going to zero when there was no rotational velocity for the vehicle.

The desired path is given as data points that have a x- and y- coordinate and the vehicle orientation. The desired velocity direction can also be specified but it can also be

determined by finding the angle between subsequent data points. The controller does not require a time variable to accompany the path that would create a specific cadence for the vehicle, instead it uses a desired velocity value that is independent of the path so the same path can be taken independent of speed without needing to use a different set of data. This can allow for the velocity to change to maximized power draw of the motors or to keep a safe speed depending on the weight of the cargo being transported. The lack of a corresponding time variable also allows for path deviations to be made by just inserting new data points into the existing path data. Given an obstacle, the control can just shift the desired path instead of having to stop and calculate a new path to execute minimize downtime.

Difficulties Encountered

When attempting to design the concept for a control scheme from scratch, there is always the difficulty in making sure everything that needs to be covered is taken care of. You have to also be aware of all possible interactions between systems so that all contingencies can be accounted for. When making program, you have to imagine and plan for any possible situation and have a proper response for each one. This is an iterative process that is never done as new situations encountered can always bring new bugs or unintended consequences. It is paramount to always err on the side of caution and safety and make sure that the default situation when something is wrong is to stop immediately.

Further Areas to Study

Some obvious areas of study are considering some of the root issues that this research has aimed to overcome such as current limitations in wireless technology.

Another added feature that can be looked into would be to develop a control scheme for multiple vehicles to come to work in synchronization to handle larger loads than the vehicles are individually designed for. This can allow for autonomous vehicles to be used for applications that would otherwise be prohibitively costly to design a purpose-built vehicle.

REFERENCES

S. Jagolinzer, S. Tosunoglu, “Trackless Multi-Dimensional Ride Vehicle,” 2015 Florida Conference on Recent Advances in Robotics, May 14-15, 2015

J. Lee, S. Choi, C. Lee, Y. Lee and K. Lee, "A study for AGV steering control and identification using vision system," Industrial Electronics, 2001. Proceedings. ISIE 2001. IEEE International Symposium on, Pusan, 2001, pp. 1575-1578 vol.3.

YuHon Tee, Y. Tan, BoonYew Teoh, EngBeng Tan and ZhenYang Wong, "A compact design of zero-radius steering autonomous amphibious vehicle with direct differential directional drive - UTAR-AAV," 2010 IEEE Conference on Robotics, Automation and Mechatronics, Singapore, 2010, pp. 176-181.

X. Ye, Z. Wu and F. Zhao, "Research on a 3 DOF Automated Guided Vehicle based on the improved feedback linearization method," Control Conference (CCC), 2014 33rd Chinese, Nanjing, 2014, pp. 184-188.

B. Xing, Wen-Jing Gao, K. Battle, T. Marwala and F. V. Nelwamondo, "Can ant algorithms make automated guided vehicle system more intelligent?," Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on, Istanbul, 2010, pp. 3226-3234.

G. Kaloutsakis, N. Tsourveloudis and P. Spanoudakis, "Design and development of an automated guided vehicle," Industrial Technology, 2003 IEEE International Conference on, 2003, pp. 990-993 Vol.2.

Xing Wu, Peihuang Lou, Dunbing Tang and Jun Yu, "An intelligent-optimal predictive controller for path tracking of Vision-based Automated Guided Vehicle," Information and Automation, 2008. ICIA 2008. International Conference on, Changsha, 2008, pp. 844-849.

M. P. Fanti, "Event control for deadlock avoidance in automated guided vehicle systems," Control Conference (ECC), 2001 European, Porto, 2001, pp. 1217-1222.

E. Gebennini, S. Dallari, A. Grassi, G. Perrica, C. Fantuzzi and R. Gamberini, "A simulation based approach for supporting Automated Guided Vehicles (AGVs) systems design," 2008 Winter Simulation Conference, Miami, FL, USA, 2008, pp. 2156-2163.

M. A. Kermanshahi, M. Rostamian and A. Vosough, "Design, production, and fuzzy control of an automated guided vehicle robot platform with capability of path following," Control, Instrumentation and Automation (ICCIA), 2011 2nd International Conference on, Shiraz, 2011, pp. 946-951.

D. Xu, Y. l. Liao, Y. j. Pang and N. Sun, "Backstepping control method for the path following for the underactuated surface vehicles," Control Conference (CCC), 2013 32nd Chinese, Xi'an, 2013, pp. 4188-4193.

Iris F.A. Vis, “Survey of research in the design and control of automated guided vehicle systems,” European Journal of Operational Research, Volume 170, Issue 3, 1 May 2006, Pages 677-709, 2006

J. Yu, P. Lou, X. Wu, “A Dual-core Real-time Embedded System for Vision-based Automated Guided Vehicle,” 2009 IITA International Conference on Control, Automation and Systems Engineering, July 11-12, 2009

J. Roberts, T. Stirling, J. Zufferey, D. Floreano, “3-D relative positioning sensor for indoor flying robots,” Autonomous Robots, 2012

S. Arora, A.K. Raina, A.K. Mittal, “Hybrid Control in Automated Guided Vehicle Systems,” 2001 IEEE Intelligent Transportation Systems Conference, August 25-29, 2001

R. Grigorevich, “Disaggregation of Control System Synthesis for Mobile Vehicle with Survivability Property,” Middle-East Journal of Scientific Research, 2013

D. Payton, "An architecture for reflexive autonomous vehicle control," Proceedings. 1986 IEEE International Conference on Robotics and Automation, 1986, pp. 1838-1845.

R. Fenton, G. Melocik and K. Olson, "On the steering of automated vehicles: Theory and experiment," in IEEE Transactions on Automatic Control, vol. 21, no. 3, pp. 306-315, Jun 1976.

M. J. Han, C. Y. Kuo and N. Y. C. Chang, "Vision-based range finder for automated guided vehicle navigation," 2016 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO), Shanghai, 2016, pp. 146-151.

S. Lee, S. Park and H. Son, "Multi-DOFs motion platform based on spherical wheels for unmanned systems," 2016 13th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), Xi'an, 2016, pp. 35-37.

K. Koganezawa and H. Yamashita, "Stiffness control of multi-DOF joint," 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, MO, 2009, pp. 363-370.

D. Mahmood, N. Javaid, U. Qasim and Z. A. Khan, "Routing Load of Route Discovery and Route Maintenance in Wireless Reactive Routing Protocols," 2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications, Victoria, BC, 2012, pp. 511-516.

V. Josef, "Trailer backing-up assistant using ultrasound sensors based control units to safely back-up the car with trailer," 2016 17th International Conference on Mechatronics - Mechatronika (ME), Prague, 2016, pp. 1-6.

S. Nanda, S. Manna, A. K. Sadhu, A. Konar and D. Bhattacharya, "Real-time surface material identification using infrared sensor to control speed of an arduino based car like mobile robot," Proceedings of the 2015 Third International Conference on Computer, Communication, Control and Information Technology (C3IT), Hooghly, 2015, pp. 1-6.

P. Yuan et al., "AGV System Based on Multi-sensor Information Fusion," 2014 International Symposium on Computer, Consumer and Control, Taichung, 2014, pp. 900-905.

C. Blum and X. Li, “Swarm Intelligence in Optimization”, ALBCOM Research Group, Universitat Polit`ecnica de Catalunya, Barcelona, Spain, School of Computer Science and Information Technology RMIT University, Melbourne, Australia

C. Cianci, X. Raemy, J. Pugh, and A. Martinoli, “Communication in a Swarm of Miniature Robots: The e-Puck as an Educational Tool for Swarm Robotics”, Swarm-Intelligent Systems Group, ÅLEcole Polytechnique FÅLedÅLerale de Lausanne, CH-1015 Lausanne, Switzerland

E. Sahin, S. Girgin, L. Bayındır, and A. Turgut; “Swarm Robotics”; KOVAN Research Laboratory Middle East Technical University, Ankara, Turkey; Team SequeL INRIA Futurs Lille, Villeneuve d’Ascq, France

F. Arvin, K. Samsudin, and A. Ramli; “Development of a Miniature Robot for Swarm Robotic Application”; International Journal of Computer and Electrical Engineering, Vol. 1, No. 4, October, 2009,1793-8163

N. Hoff, R. Wood, and R. Nagpal. "Distributed colony-level algorithm switching for robot swarm foraging." Distributed Autonomous Robotic Systems (2013): 417-430.

J. Hecker, and M. Moses, 2015, Beyond Pheromones: Evolving Error-Tolerant, Flexible, and Scalable Ant-Inspired Robot Swarms

L. Parker, “Multiple Mobile Robot System”, Springer-Verlag Berlin Heidelberg, Springer Berlin Heidelberg 2008

M. Ling, J. Chen, Z. Li, and D. Zhang. "Relative Localization Method of Multiple Micro Robots Based on Simple Sensors." Int J Adv Robotic Sy International Journal of Advanced Robotic Systems (2013): 1. Web. 13 Feb. 2016.

S. Majercik. "Swarm-Based Path Creation in Dynamic Environments for Search and Rescue." Proc. of Genetic and Evolutionary Computation Conference, Philadelphia, PA, USA. N.p., 11 July 2012. Web. 12 Feb. 2016.

K. Stolleis "The Ant and the Trap: Evolution of Ant-Inspired Obstacle Avoidance in a Multi-Agent Robotic System." 26 Jun. 2015.

T. Shaneyfelt, M. Joordens, K. Nagothu, J. Prevost, A. Kumar, M. Ghazi, M. Jamshidi, 2008, Control and simulation of robotic swarms in heterogeneous environments, in SMC 2008: Proceedings of 2008 IEEE International Conference on Systems, Man and Cybernetics, IEEE, Piscataway, N.J., pp. 1314-1319

“Automatic Guided Vehicles (AGV) Drive & Steering Options” Transbotics, www.transbotics.com/learning-center/drive-steering

“Laser Target Navigation” Applied & Integrated Manufacturing Inc. (AIM) http://www.aimagv.com/laser-target.html

“Ensor- Egemin Navigation System on Robot” Engemin Automation www.egeminusa.com/automated-guided-vehicles/software/ensor/

Appendix 1: Data from Vehicle Loop

Starting on the following page is a printout of the desired control of a vehicle following a predetermined path. It shows the desired values and controls on the macro level then breaks it down to a per wheel output that makes up the 8 values that are sent to the 8 motors that control the motion and orientation of the platform.

Desired Path				Vehicle ratio		Signal					Wheel 1		Wheel 2		Wheel 3		Wheel 4		Output				Error				
X	Y	DIR	ORI			Time	L VEI	DIR	R VEL	ORI	Vel	Theta	Vel	Theta	Vel	Theta	Vel	Theta	X	Y	DIR	ORI	X	Y	XY	DIR	ORI
0.00	0.00	0.00	0.00	x	3.50	0	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1.00	0.00	0.00	0.00	y	1.00	0.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2.00	0.00	0.00	0.00			0.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
3.00	0.00	0.00	0.00	theta1	15.95	0.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	1.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4.00	0.00	0.00	0.00	phi1	105.95	0.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	2.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5.00	0.00	0.00	0.00			0.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	2.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6.00	0.00	0.00	0.00	theta2	164.05	0.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	3.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
7.00	0.00	0.00	0.00	phi2	254.05	0.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	3.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8.00	0.00	0.00	0.00			0.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	4.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9.00	0.00	0.00	0.00	theta3	195.95	0.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	4.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10.00	0.00	0.00	0.00	phi3	285.95	1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11.00	0.00	0.00	0.00			1.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
12.00	0.00	0.00	0.00	theta4	344.05	1.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	6.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
13.00	0.00	0.00	0.00	phi4	74.05	1.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	6.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
14.00	0.00	0.00	0.00			1.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	7.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15.00	0.00	0.00	0.00	ROT GAI	3.00	1.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	7.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
16.00	0.00	0.00	0.00	DIR GAI	1.00	1.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	8.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
17.00	0.00	0.00	0.00	YD GAIN	0.20	1.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	8.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18.00	0.00	0.00	0.00	XD GAIN	0.20	1.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	9.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
19.00	0.00	0.00	0.00			1.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	9.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20.00	0.00	0.00	0.00			2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	10.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
21.00	0.00	0.00	0.00			2.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	10.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
22.00	0.00	0.00	0.00			2.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	11.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
23.00	0.00	0.00	0.00			2.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	11.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
24.00	0.00	0.00	0.00			2.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	12.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25.00	0.00	0.00	0.00			2.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	12.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
26.00	0.00	0.00	0.00			2.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	13.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
27.00	0.00	0.00	0.00			2.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	13.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
28.00	0.00	0.00	0.00			2.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	14.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
29.00	0.00	0.00	0.00			2.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	14.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
30.00	0.00	0.00	0.00			3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	15.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
31.00	0.00	0.00	0.00			3.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	15.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
32.00	0.00	0.00	0.00			3.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	16.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
33.00	0.00	0.00	0.00			3.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	16.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
34.00	0.00	0.00	0.00			3.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	17.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
35.00	0.00	0.00	0.00			3.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	17.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
36.00	0.00	0.00	0.00			3.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	18.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
37.00	0.00	0.00	0.00			3.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	18.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
38.00	0.00	0.00	0.00			3.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	19.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
39.00	0.00	0.00	0.00			3.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	19.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
40.00	0.00	0.00	0.00			4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	20.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
41.00	0.00	0.00	0.00			4.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	20.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
42.00	0.00	0.00	0.00			4.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	21.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
43.00	0.00	0.00	0.00			4.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	21.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
44.00	0.00	0.00	0.00			4.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	22.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
45.00	0.00	0.00	0.00			4.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	22.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
46.00	0.00	0.00	0.00			4.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	23.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
47.00	0.00	0.00	0.00			4.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	23.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
48.00	0.00	0.00	0.00			4.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	24.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
49.00	0.00	0.00	0.00			4.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	24.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
50.00	0.00	0.00	0.00			5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	25.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
51.00	0.00	0.00	0.00			5.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	25.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
52.00	0.00	0.00	0.00			5.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	26.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
53.00	0.00	0.00	0.00			5.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	26.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
54.00	0.00	0.00	0.00			5.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	27.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
55.00	0.00	0.00	0.00			5.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	27.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
56.00	0.00	0.00	0.00			5.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	28.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
57.00	0.00	0.00	0.00			5.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	28.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
58.00	0.00	0.00	0.00			5.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	29.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
59.00	0.00	0.00	0.00			5.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	29.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
60.00	0.00	0.00	0.00			6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	30.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
61.00	0.00	0.00	0.00			6.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	30.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
62.00	0.00	0.00	0.00			6.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	31.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
63.00	0.00	0.00	0.00			6.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	31.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
64.00	0.00	0.00	0.00			6.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	32.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
65.00	0.00	0.00	0.00			6.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	32.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
66.00	0.00	0.00	0.00			6.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	33.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
67.00	0.00	0.00	0.00			6.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	33.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
68.00	0.00	0.00	0.00			6.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	34.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
69.00	0.00	0.00	0.00			6.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	34.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
70.00	0.00	0.00	0.00			7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	35.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
71.00	0.00	0.00	0.00			7.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	35.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
72.00	0.00	0.00	0.00			7.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	36.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
73.00	0.00	0.00	0.00			7.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	36.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
74.00	0.00	0.00	0.00			7.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	37.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
75.00	0.00	0.00	0.00			7.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	37.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
76.00	0.00	0.00	0.00			7.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	38.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
77.00	0.00	0.00	0.00			7.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	38.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
78.00	0.00	0.00	0.00			7.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	39.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
79.00	0.00	0.00	0.00			7.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	39.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
80.00	0.00	0.00	0.00			8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
81.00	0.00	0.00	0.00			8.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	40.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
82.00	0.00	0.00	0.00			8.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	41.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
83.00	0.00	0.00	0.00			8.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	41.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
84.00	0.00	0.00	0.00			8.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	42.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
85.00	0.00	0.00	0.00			8.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	42.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
86.00	0.00	0.00	0.00			8.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	43.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
87.00	0.00	0.00	0.00			8.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	43.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
88.00	0.00	0.00	0.00			8.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	44.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
89.00	0.00	0.00	0.00			8.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	44.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
90.00	0.00	0.00	0.00			9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	45.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
91.00	0.00	0.00	0.00			9.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	45.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
92.00	0.00	0.00	0.00			9.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	46.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
93.00	0.00	0.00	0.00			9.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	46.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
94.00	0.00	0.00	0.00			9.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	47.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
95.00	0.00	0.00	0.00			9.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	47.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
96.00	0.00	0.00	0.00			9.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	48.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
97.00	0.00	0.00	0.00			9.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	48.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
98.00	0.00	0.00	0.00			9.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	49.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
99.00	0.00	0.00	0.00			9.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	49.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
100.00	0.00	0.00	0.00			10	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	50.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
101.00	0.01	1.15	1.15			10.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	50.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
102.00	0.04	2.29	2.29			10.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	51.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
103.00	0.09	3.44	3.44			10.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	51.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
104.00	0.16	4.59	4.59			10.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	52.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
105.00	0.25	5.74	5.74			10.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	52.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
106.00	0.36	6.89	6.89			10.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	53.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
107.00	0.49	8.05	8.05			10.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	53.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
108.00	0.64	9.21	9.21			10.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	54.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
109.00	0.82	10.37	10.37			10.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	54.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
110.00	1.01	11.54	11.54			11	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	55.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
111.00	1.23	12.71	12.71			11.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	55.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
112.00	1.46	13.89	13.89			11.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	56.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
113.00	1.72	15.07	15.07			11.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	56.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
114.00	2.00	16.26	16.26			11.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	57.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
115.00	2.30	17.46	17.46			11.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	57.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
116.00	2.63	18.66	18.66			11.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	58.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

117.00	2.98	19.88	19.88	11.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	58.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
118.00	3.35	21.10	21.10	11.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	59.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
119.00	3.75	22.33	22.33	11.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	59.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
120.00	4.17	23.58	23.58	12	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	60.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
121.00	4.62	24.83	24.83	12.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	60.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
122.00	5.10	26.10	26.10	12.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	61.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
123.00	5.60	27.39	27.39	12.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	61.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
124.00	6.14	28.69	28.69	12.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	62.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
125.00	6.70	30.00	30.00	12.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	62.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
126.00	7.29	31.33	31.33	12.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	63.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
127.00	7.92	32.68	32.68	12.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	63.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
128.00	8.58	34.06	34.06	12.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	64.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
129.00	9.27	35.45	35.45	12.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	64.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
130.00	10.00	36.87	36.87	13	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	65.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
131.00	10.77	38.32	38.32	13.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	65.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
132.00	11.58	39.79	39.79	13.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	66.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
133.00	12.44	41.30	41.30	13.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	66.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
134.00	13.34	42.84	42.84	13.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	67.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
135.00	14.29	44.43	44.43	13.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	67.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
136.00	15.30	46.05	46.05	13.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	68.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
137.00	16.37	47.73	47.73	13.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	68.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
138.00	17.50	49.46	49.46	13.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	69.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
139.00	18.71	51.26	51.26	13.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	69.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
140.00	20.00	53.13	53.13	14	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	70.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
141.00	21.38	55.08	55.08	14.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	70.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
142.00	22.87	57.14	57.14	14.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	71.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
143.00	24.49	59.32	59.32	14.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	71.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
144.00	26.25	61.64	61.64	14.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	72.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
145.00	28.21	64.16	64.16	14.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	72.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
146.00	30.40	66.93	66.93	14.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	73.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
147.00	32.94	70.05	70.05	14.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	73.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
148.00	36.00	73.74	73.74	14.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	74.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
149.00	40.05	78.52	78.52	14.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	74.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
150.00	50.00	90.00	90.00	15	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	75.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
149.00	59.95	101.48	101.48	15.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	75.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
148.00	64.00	106.26	106.26	15.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	76.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
147.00	67.06	109.95	109.95	15.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	76.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
146.00	69.60	113.07	113.07	15.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	77.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
145.00	71.79	115.84	115.84	15.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	77.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
144.00	73.75	118.36	118.36	15.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	78.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
143.00	75.51	120.68	120.68	15.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	78.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
142.00	77.13	122.86	122.86	15.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	79.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
141.00	78.62	124.92	124.92	15.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	79.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
140.00	80.00	126.87	126.87	16	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	80.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
139.00	81.29	128.74	128.74	16.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	80.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
138.00	82.50	130.54	130.54	16.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	81.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
137.00	83.63	132.27	132.27	16.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	81.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
136.00	84.70	133.95	133.95	16.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	82.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
135.00	85.71	135.57	135.57	16.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	82.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
134.00	86.66	137.16	137.16	16.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	83.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
133.00	87.56	138.70	138.70	16.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	83.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
132.00	88.42	140.21	140.21	16.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	84.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
131.00	89.23	141.68	141.68	16.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	84.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
130.00	90.00	143.13	143.13	17	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	85.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
129.00	90.73	144.55	144.55	17.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	85.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
128.00	91.42	145.94	145.94	17.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	86.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
127.00	92.08	147.32	147.32	17.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	86.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
126.00	92.71	148.67	148.67	17.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	87.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
125.00	93.30	150.00	150.00	17.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	87.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
124.00	93.86	151.31	151.31	17.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	88.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
123.00	94.40	152.61	152.61	17.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	88.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
122.00	94.90	153.90	153.90	17.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	89.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
121.00	95.38	155.17	155.17	17.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	89.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
120.00	95.83	156.42	156.42	18	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	90.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
119.00	96.25	157.67	157.67	18.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	90.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
118.00	96.65	158.90	158.90	18.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	91.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
117.00	97.02	160.12	160.12	18.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	91.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
116.00	97.37	161.34	161.34	18.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	92.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
115.00	97.70	162.54	162.54	18.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	92.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
114.00	98.00	163.74	163.74	18.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	93.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
113.00	98.28	164.93	164.93	18.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	93.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
112.00	98.54	166.11	166.11	18.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	94.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
111.00	98.77	167.29	167.29	18.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	94.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
110.00	98.99	168.46	168.46	19	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	95.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
109.00	99.18	169.63	169.63	19.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	95.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
108.00	99.36	170.79	170.79	19.2	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	96.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
107.00	99.51	171.95	171.95	19.3	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	96.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
106.00	99.64	173.11	173.11	19.4	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	97.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
105.00	99.75	174.26	174.26	19.5	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	97.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
104.00	99.84	175.41	175.41	19.6	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	98.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
103.00	99.91	176.56	176.56	19.7	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	98.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
102.00	99.96	177.71	177.71	19.8	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	99.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
101.00	99.99	178.85	178.85	19.9	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	99.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
100.00	100.00	180.00	180.00	20	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	100.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
99.00	100.00	180.00	180.00	20.1	5.00	0.00	0.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	5.00	0.00	100.50	0.00	0.00	0.00	0.00	0.00	0.00	0.57	0.57
98.00	100.00	180.00	180.00	20.2	5.00	0.57	1.72	0.00	4.84	20.62	4.80	-19.50	5.70	-16.33	5.73	17.29	101.00	0.00	0.00	0.00	0.00	0.01	0.01	1.15	1.15
97.00	100.00	180.00	180.00	20.3	5.00	1.72	3.44	0.17	5.32	40.34	5.15	-38.05	6.74	-28.09	6.87	30.08	101.50	0.00	0.57	0.17	0.00	0.02	0.02	1.15	1.55
96.00	100.00	180.00	180.00	20.4	5.00	2.86	4.64	0.52	5.97	51.45	5.66	-48.87	7.58	-34.19	7.82	36.67	102.00	0.02	1.72	0.52	0.00	0.02	0.02	0.58	1.78
95.00	100.00	180.00	180.00	20.5	5.00	3.43	5.33	0.98	6.40	56.53	6.05	-54.29	8.12	-37.25	8.38	39.58	102.50	0.04	2.86	0.98	0.00	0.02	0.02	0.01	1.89
94.00	100.00	180.00	180.00	20.6	5.00	3.43	5.66	1.51	6.58	58.45	6.30	-56.86	8.41	-38.83	8.62	40.56	103.00	0.07	3.43	1.51	0.00	0.02	0.02	0.01	1.93
93.00	100.00	180.00	180.00	20.7	5.00	3.44	5.78	2.08	6.62	58.99	6.42	-57.90	8.54	-39.54	8.69	40.75	103.50	0.10	3.43	2.08	0.00	0.02	0.02	0.58	1.93
92.00	100.00	180.00	180.00	20.8	5.00	4.01	5.80	2.66	6.64	59.13	6.43	-58.05	8.56	-39.63	8.71	40.83	104.00	0.13	3.44	2.66	0.00	0.03	0.03	1.15	1.93
91.00	100.00	180.00	180.00	20.9	5.00	5.16	5.79	3.24	6.67	59.27	6.38	-57.73	8.52	-39.33	8.73	41.03	104.50	0.17	4.01	3.24	0.00	0.03	0.03	1.15	1.92
90.00	100.00	180.00	180.00	21	5.00	6.30	5.77	3.82	6.70	59.37	6.33	-57.37	8.47	-39.00	8.75	41.20	104.99	0.21	5.16	3.82	0.00	0.04	0.04	0.58	1.92
89.00	100.00	180.00	180.00	21.1	5.00	6.86	5.75	4.39	6.68	59.26	6.32	-57.26	8.45	-38.94	8.73	41.14	105.49	0.27	6.30	4.39	0.00	0.03	0.03	0.01	1.91
88.00	100.00	180.00	180.00	21.2	5.00	6.87	5.74	4.97	6.63	58.97	6.35	-57.42	8.48	-39.16	8.69	40.85	105.99	0.33	6.86	4.97	0.00	0.03	0.03	0.01	1.91
87.00	100.00	180.00	180.00	21.3	5.00	6.88	5.73	5.54	6.59	58.70	6.39	-57.62	8.51	-39.39	8.66	40.58	106.48	0.39	6.87	5.54	0.00	0.03	0.03	0.58	1.91
86.00	100.00	180.00	180.00	21.4	5.00	7.45	5.73	6.11	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	106.98	0.45	6.88	6.11	0.00	0.04	0.04	1.15	1.91
85.00	100.00	180.00	180.00	21.5	5.00	8.59	5.73	6.69	6.63	58.92	6.35	-57.37	8.47	-39.12	8.68	40.82	107.48	0.51	7.45	6.69	0.00	0.05	0.05	1.15	1.91
84.00	100.00	180.00	180.00	21.6	5.00	9.73	5.73	7.26	6.67	59.15	6.30	-57.14	8.44	-38.87	8.72	41.08	107.97	0.59	8.59	7.26	0.00	0.05	0.05	0.58	1.91
83.00	100.00	180.00	180.00	21.7	5.00	10.30	5.73	7.83	6.67	59.15	6.31	-57.14	8.44	-38.87	8.72	41.07	108.46	0.67	9.73	7.83	0.00	0.05	0.05	0.02	1.91
82.00	100.00	180.00	180.00	21.8	5.00	10.30	5.73	8.41	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	108.95	0.76	10.30	8.41	0.00	0.05	0.05	0.02	1.91
81.00	100.00	180.00	180.00	21.9	5.00	10.31	5.73	8.98	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	109.45	0.85	10.30	8.98	0.00	0.05	0.05	0.59	1.91
80.00	100.00	180.00	180.00	22	5.00	10.89	5.73	9.55	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	109.94	0.94	10.31	9.55	0.00	0.06	0.06	1.15	1.91
79.00	100.00	180.00	180.00	22.1	5.00	12.03	5.73	10.13	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.83	110.43	1.04	10.89	10.13	0.00	0.06	0.06	1.15	1.91
78.00	100.00	180.00	180.00	22.2	5.00	13.16	5.73	10.70	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.08	110.92	1.14	12.03	10.70	0.00	0.07	0.07	0.58	1.91
77.00	100.00	180.00	180.00	22.3	5.00	13.73	5.73	11.27	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	111.41	1.25	13.16	11.27	0.00	0.06	0.06	0.02	1.91
76.00	100.00	180.00	180.00	22.4	5.00	13.74	5.73	11.85	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	111.89	1.37	13.73	11.85	0.00	0.06	0.06	0.03	1.91
75.00	100.00	180.00	180.00	22.5	5.00	13.75	5.73	12.42	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	112.38	1.49	13.74	12.42	0.00	0.06	0.06	0.59	1.91
74.00	100.00	180.00	180.00	22.6	5.00	14.33	5.73	12.99	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	112.86	1.61	13.75	12.99	0.00	0.07	0.07	1.15	1.91
73.00	100.00	180.00	180.00	22.7	5.00	15.47	5.73	13.57	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.83	113.35	1.73	14.33	13.57	0.00	0.08	0.08	1.15	1.91
72.00	100.00	180.00	180.00	22.8	5.00	16.60	5.73	14.14	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.08	113.83	1.87	15.47	14.14	0.00	0.08	0.08	0.58	1.91
71.00	100.00	180.00	180.00	22.9	5.00	17.17	5.73	14.71	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	114.31	2.01	16.60	14.71	0.00	0.08	0.08	0.02	1.91
70.00	100.00	180.00	180.00	23	5.00	17.17	5.73	15.29	6.63	58.91	6.35	-57.38	8.47	-39.13	8.68	40.82	114.79	2.16	17.17	15.29	0.00	0.08	0.08	0.03	1.91
69.00	100.00	180.00	180.00	23.1	5.00	17.19	5.73	15.86	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	115.26	2.31	17.17	15.86	0.00	0.08	0.08	0.60	1.91
68.00	100.00	180.00	180.00	23.2	5.00	17.77	5.73	16.43	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	115.74	2.45	17.19	16.43	0.00	0.09	0.09	1.15	1.91
67.00	100.00	180.00	180.00	23.3	5.00	18.91	5.73	17.01	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.82	116.22	2.61	17.77	17.01	-0.01	0.09	0.09	1.15	1.91
66.00	100.00	180.00	180.00	23.4	5.00	20.04	5.73	17.58	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	116.69	2.77	18.91	17.58	-0.01	0.10	0.10	0.58	1.91
65.00	100.00	180.00	180.00	23.5	5.00	20.60	5.73	18.15	6.67	59.14	6.31	-57.15	8.44	-38.88	8.71	41.07	117.16	2.94	20.04	18.15	-0.01	0.10	0.10	0.03	1.91

64.00	100.00	180.00	180.00	23.6	5.00	20.61	5.73	18.73	6.63	58.91	6.35	-57.38	8.47	-39.13	8.68	40.82	117.63	3.12	20.60	18.73	-0.01	0.09	0.09	0.04	1.91
63.00	100.00	180.00	180.00	23.7	5.00	20.63	5.73	19.30	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	118.10	3.29	20.61	19.30	-0.01	0.10	0.10	0.60	1.91
62.00	100.00	180.00	180.00	23.8	5.00	21.21	5.73	19.87	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	118.56	3.47	20.63	19.87	-0.01	0.10	0.10	1.15	1.91
61.00	100.00	180.00	180.00	23.9	5.00	22.35	5.73	20.45	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.82	119.03	3.65	21.21	20.45	-0.01	0.11	0.11	1.15	1.91
60.00	100.00	180.00	180.00	24	5.00	23.47	5.73	21.02	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	119.49	3.84	22.35	21.02	-0.01	0.11	0.11	0.58	1.91
59.00	100.00	180.00	180.00	24.1	5.00	24.04	5.73	21.59	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	119.95	4.04	23.47	21.59	-0.01	0.11	0.11	0.03	1.91
58.00	100.00	180.00	180.00	24.2	5.00	24.05	5.73	22.17	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.82	120.41	4.24	24.04	22.17	-0.01	0.11	0.11	0.04	1.91
57.00	100.00	180.00	180.00	24.3	5.00	24.07	5.73	22.74	6.59	58.68	6.39	-57.60	8.50	-39.38	8.65	40.57	120.86	4.44	24.05	22.74	-0.01	0.11	0.11	0.60	1.91
56.00	100.00	180.00	180.00	24.4	5.00	24.65	5.73	23.31	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	121.32	4.65	24.07	23.31	-0.01	0.12	0.12	1.16	1.91
55.00	100.00	180.00	180.00	24.5	5.00	25.79	5.73	23.89	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	121.77	4.86	24.65	23.89	-0.02	0.13	0.13	1.15	1.91
54.00	100.00	180.00	180.00	24.6	5.00	26.91	5.73	24.46	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	122.23	5.07	25.79	24.46	-0.02	0.13	0.13	0.58	1.91
53.00	100.00	180.00	180.00	24.7	5.00	27.47	5.73	25.03	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	122.67	5.30	26.91	25.03	-0.02	0.13	0.13	0.03	1.91
52.00	100.00	180.00	180.00	24.8	5.00	27.48	5.73	25.61	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.81	123.11	5.53	27.47	25.61	-0.01	0.12	0.13	0.04	1.91
51.00	100.00	180.00	180.00	24.9	5.00	27.51	5.73	26.18	6.59	58.68	6.39	-57.60	8.50	-39.38	8.65	40.57	123.56	5.76	27.48	26.18	-0.02	0.13	0.13	0.61	1.91
50.00	100.00	180.00	180.00	25	5.00	28.09	5.73	26.75	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	124.00	5.99	27.51	26.75	-0.02	0.13	0.13	1.16	1.91
49.00	100.00	180.00	181.80	25.1	5.00	29.22	5.73	27.32	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	124.44	6.23	28.09	27.32	-0.02	0.14	0.14	1.15	1.91
48.00	100.00	180.00	183.60	25.2	5.00	30.35	5.73	27.90	6.67	59.14	6.31	-57.15	8.44	-38.88	8.71	41.07	124.88	6.47	29.22	27.90	-0.02	0.14	0.14	0.59	1.91
47.00	100.00	180.00	185.40	25.3	5.00	30.91	5.73	28.47	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	125.31	6.73	30.35	28.47	-0.02	0.14	0.14	0.04	1.91
46.00	100.00	180.00	187.20	25.4	5.00	30.92	5.73	29.04	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.81	125.74	6.98	30.91	29.04	-0.02	0.14	0.14	0.05	1.91
45.00	100.00	180.00	189.00	25.5	5.00	30.94	5.73	29.62	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	126.17	7.24	30.92	29.62	-0.02	0.14	0.14	0.61	1.91
44.00	100.00	180.00	190.80	25.6	5.00	31.53	5.73	30.19	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	126.60	7.50	30.94	30.19	-0.03	0.15	0.15	1.16	1.91
43.00	100.00	180.00	192.60	25.7	5.00	32.66	5.73	30.76	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	127.02	7.76	31.53	30.76	-0.03	0.15	0.16	1.15	1.91
42.00	100.00	180.00	194.40	25.8	5.00	33.78	5.73	31.34	6.67	59.14	6.31	-57.15	8.44	-38.88	8.71	41.07	127.44	8.03	32.66	31.34	-0.03	0.16	0.16	0.59	1.91
41.00	100.00	180.00	196.20	25.9	5.00	34.34	5.73	31.91	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	127.86	8.31	33.78	31.91	-0.03	0.16	0.16	0.04	1.91
40.00	100.00	180.00	198.00	26	5.00	34.36	5.73	32.48	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.81	128.27	8.59	34.34	32.48	-0.03	0.15	0.16	0.05	1.91
39.00	100.00	180.00	199.80	26.1	5.00	34.38	5.73	33.06	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	128.69	8.87	34.36	33.06	-0.03	0.16	0.16	0.61	1.91
38.00	100.00	180.00	201.60	26.2	5.00	34.97	5.73	33.63	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	129.10	9.15	34.38	33.63	-0.03	0.16	0.17	1.16	1.91
37.00	100.00	180.00	203.40	26.3	5.00	36.10	5.73	34.20	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	129.51	9.44	34.97	34.20	-0.04	0.17	0.17	1.15	1.91
36.00	100.00	180.00	205.20	26.4	5.00	37.22	5.73	34.78	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	129.91	9.73	36.10	34.78	-0.04	0.17	0.18	0.59	1.91
35.00	100.00	180.00	207.00	26.5	5.00	37.78	5.73	35.35	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	130.31	10.04	37.22	35.35	-0.04	0.17	0.17	0.04	1.91
34.00	100.00	180.00	208.80	26.6	5.00	37.80	5.73	35.92	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	130.71	10.34	37.78	35.92	-0.04	0.17	0.17	0.05	1.91
33.00	100.00	180.00	210.60	26.7	5.00	37.82	5.73	36.50	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	131.10	10.65	37.80	36.50	-0.04	0.17	0.17	0.61	1.91
32.00	100.00	180.00	212.40	26.8	5.00	38.41	5.73	37.07	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	131.50	10.96	37.82	37.07	-0.04	0.18	0.18	1.16	1.91
31.00	100.00	180.00	214.20	26.9	5.00	39.54	5.73	37.64	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	131.89	11.27	38.41	37.64	-0.05	0.18	0.19	1.15	1.91
30.00	100.00	180.00	216.00	27	5.00	40.66	5.73	38.22	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	132.27	11.58	39.54	38.22	-0.05	0.19	0.19	0.59	1.91
29.00	100.00	180.00	217.80	27.1	5.00	41.22	5.73	38.79	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	132.65	11.91	40.66	38.79	-0.05	0.18	0.19	0.04	1.91
28.00	100.00	180.00	219.60	27.2	5.00	41.23	5.73	39.36	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	133.03	12.24	41.22	39.36	-0.05	0.18	0.19	0.05	1.91
27.00	100.00	180.00	221.40	27.3	5.00	41.26	5.73	39.94	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	133.40	12.57	41.23	39.94	-0.05	0.18	0.19	0.61	1.91
26.00	100.00	180.00	223.20	27.4	5.00	41.85	5.73	40.51	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	133.78	12.90	41.26	40.51	-0.05	0.19	0.20	1.16	1.91
25.00	100.00	180.00	225.00	27.5	5.00	42.98	5.73	41.08	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	134.15	13.23	41.85	41.08	-0.06	0.20	0.20	1.15	1.91
24.00	100.00	180.00	226.80	27.6	5.00	44.10	5.73	41.66	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	134.52	13.57	42.98	41.66	-0.06	0.20	0.21	0.59	1.91
23.00	100.00	180.00	228.60	27.7	5.00	44.66	5.73	42.23	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	134.88	13.92	44.10	42.23	-0.06	0.20	0.20	0.04	1.91
22.00	100.00	180.00	230.40	27.8	5.00	44.67	5.73	42.80	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	135.23	14.27	44.66	42.80	-0.06	0.20	0.20	0.06	1.91
21.00	100.00	180.00	232.20	27.9	5.00	44.70	5.73	43.38	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	135.59	14.62	44.67	43.38	-0.06	0.20	0.21	0.61	1.91
20.00	100.00	180.00	234.00	28	5.00	45.29	5.73	43.95	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	135.94	14.98	44.70	43.95	-0.06	0.20	0.21	1.16	1.91
19.00	100.00	180.00	235.80	28.1	5.00	46.42	5.73	44.52	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	136.30	15.33	45.29	44.52	-0.07	0.21	0.22	1.15	1.91
18.00	100.00	180.00	237.60	28.2	5.00	47.54	5.73	45.10	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	136.64	15.69	46.42	45.10	-0.07	0.21	0.22	0.59	1.91
17.00	100.00	180.00	239.40	28.3	5.00	48.10	5.73	45.67	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	136.98	16.06	47.54	45.67	-0.07	0.21	0.22	0.04	1.91
16.00	100.00	180.00	241.20	28.4	5.00	48.11	5.73	46.24	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	137.31	16.43	48.10	46.24	-0.07	0.21	0.22	0.06	1.91
15.00	100.00	180.00	243.00	28.5	5.00	48.14	5.73	46.82	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	137.65	16.81	48.11	46.82	-0.07	0.21	0.22	0.61	1.91
14.00	100.00	180.00	244.80	28.6	5.00	48.73	5.73	47.39	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	137.98	17.18	48.14	47.39	-0.07	0.22	0.23	1.16	1.91
13.00	100.00	180.00	246.60	28.7	5.00	49.86	5.73	47.96	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	138.31	17.55	48.73	47.96	-0.08	0.22	0.23	1.15	1.91
12.00	100.00	180.00	248.40	28.8	5.00	50.98	5.73	48.54	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	138.63	17.94	49.86	48.54	-0.08	0.22	0.24	0.59	1.91
11.00	100.00	180.00	250.20	28.9	5.00	51.54	5.73	49.11	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	138.95	18.33	50.98	49.11	-0.08	0.22	0.24	0.04	1.91
10.00	100.00	180.00	252.00	29	5.00	51.55	5.73	49.68	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	139.26	18.72	51.54	49.68	-0.08	0.22	0.23	0.06	1.91
9.00	100.00	180.00	253.80	29.1	5.00	51.58	5.73	50.25	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	139.57	19.11	51.55	50.25	-0.08	0.22	0.24	0.61	1.91
8.00	100.00	180.00	255.60	29.2	5.00	52.16	5.73	50.83	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	139.88	19.50	51.58	50.83	-0.09	0.23	0.24	1.16	1.91
7.00	100.00	180.00	257.40	29.3	5.00	53.30	5.73	51.40	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	140.19	19.90	52.16	51.40	-0.09	0.23	0.25	1.15	1.91
6.00	100.00	180.00	259.20	29.4	5.00	54.42	5.73	51.97	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	140.48	20.30	53.30	51.97	-0.09	0.23	0.25	0.59	1.91
5.00	100.00	180.00	261.00	29.5	5.00	54.98	5.73	52.55	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	140.78	20.70	54.42	52.55	-0.09	0.23	0.25	0.04	1.91
4.00	100.00	180.00	262.80	29.6	5.00	54.99	5.73	53.12	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	141.06	21.11	54.98	53.12	-0.09	0.23	0.25	0.05	1.91
3.00	100.00	180.00	264.60	29.7	5.00	55.02	5.73	53.69	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	141.35	21.52	54.99	53.69	-0.09	0.23	0.25	0.61	1.91
2.00	100.00	180.00	266.40	29.8	5.00	55.60	5.73	54.27	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	141.64	21.93	55.02	54.27	-0.10	0.24	0.26	1.16	1.91
1.00	100.00	180.00	268.20	29.9	5.00	56.74	5.73	54.84	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	141.92	22.34	55.60	54.84	-0.10	0.24	0.26	1.15	1.91
0.00	100.00	180.00	270.00	30	5.00	57.86	5.73	55.41	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	142.19	22.76	56.74	55.41	-0.11	0.24	0.27	0.59	1.91
-1.00	100.00	180.00	271.80	30.1	5.00	58.42	5.73	55.99	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	142.46	23.19	57.86	55.99	-0.10	0.24	0.26	0.04	1.91
-2.00	100.00	180.00	273.60	30.2	5.00	58.43	5.73	56.56	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	142.72	23.61	58.42	56.56	-0.10	0.24	0.26	0.05	1.91
-3.00	100.00	180.00	275.40	30.3	5.00	58.46	5.73	57.13	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	142.98	24.04	58.43	57.13	-0.10	0.24	0.27	0.61	1.91
-4.00	100.00	180.00	277.20	30.4	5.00	59.04	5.73	57.71	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	143.24	24.46	58.46	57.71	-0.11	0.25	0.27	1.16	1.91
-5.00	100.00	180.00	279.00	30.5	5.00	60.18	5.73	58.28	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	143.50	24.89	59.04	58.28	-0.12	0.25	0.28	1.15	1.91
-6.00	100.00	180.00	280.80	30.6	5.00	61.30	5.73	58.85	6.67	59.14	6.31	-57.16	8.44	-38.88	8.71	41.07	143.75	25.33	60.18	58.85	-0.12	0.25	0.28	0.59	1.91
-7.00	100.00	180.00	282.60	30.7	5.00	61.86	5.73	59.43	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	143.99	25.76	61.30	59.43	-0.12	0.25	0.28	0.04	1.91
-8.00	100.00	180.00	284.40	30.8	5.00	61.87	5.73	60.00	6.63	58.91	6.35	-57.39	8.47	-39.14	8.68	40.81	144.23	26.21	61.86	60.00	-0.12	0.25	0.28	0.05	1.91
-9.00	100.00	180.00	286.20	30.9	5.00	61.90	5.73	60.57	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	144.46	26.65	61.87	60.57	-0.12	0.25	0.28	0.61	1.91
-10.00	100.00	180.00	288.00	31	5.00	62.48	5.73	61.15	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	144.70	27.09	61.90	61.15	-0.12	0.26	0.29	1.16	1.91
-11.00	100.00	180.00	289.80	31.1	5.00	63.62	5.73	61.72	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	144.93	27.53	62.48	61.72	-0.13	0.26	0.29	1.15	1.91
-12.00	100.00	180.00	291.60	31.2	5.00	64.74	5.73	62.29	6.67	59.14	6.31	-57.15	8.44	-38.88	8.71	41.07	145.15	27.98	63.62	62.29	-0.13	0.26	0.29	0.59	1.91
-13.00	100.00	180.00	293.40	31.3	5.00	65.30	5.73	62.87	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.06	145.36	28.43	64.74	62.87	-0.13	0.26	0.29	0.04	1.91
-14.00	100.00	180.00	295.20	31.4	5.00	65.31	5.73	63.44	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.81	145.57	28.89	65.30	63.44	-0.13	0.26	0.29	0.05	1.91
-15.00	100.00	180.00	297.00	31.5	5.00	65.34	5.73	64.01	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	145.78	29.34	65.31	64.01	-0.13	0.26	0.29	0.61	1.91
-16.00	100.00	180.00	298.80	31.6	5.00	65.92	5.73	64.59	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	145.99	29.79	65.34	64.59	-0.14	0.27	0.30	1.16	1.91
-17.00	100.00	180.00	300.60	31.7	5.00	67.06	5.73	65.16	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	146.19	30.25	65.92	65.16	-0.15	0.27	0.31	1.15	1.91
-18.00	100.00	180.00	302.40	31.8	5.00	68.18	5.73	65.73	6.67	59.14	6.31	-57.15	8.44	-38.88	8.71	41.07	146.39	30.71	67.06	65.73	-0.15	0.27	0.31	0.59	1.91
-19.00	100.00	180.00	304.20	31.9	5.00	68.74	5.73	66.31	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	146.58	31.17	68.18	66.31	-0.15	0.27	0.31	0.04	1.91
-20.00	100.00	180.00	306.00	32	5.00	68.76	5.73	66.88	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.81	146.76	31.64	68.74	66.88	-0.14	0.27	0.31	0.05	1.91
-21.00	100.00	180.00	307.80	32.1	5.00	68.78	5.73	67.45	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	146.94	32.11	68.76	67.45	-0.15	0.27	0.31	0.61	1.91
-22.00	100.00	180.00	309.60	32.2	5.00	69.36	5.73	68.03	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	147.12	32.57	68.78	68.03	-0.15	0.27	0.31	1.16	1.91
-23.00	100.00	180.00	311.40	32.3	5.00	70.49	5.73	68.60	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	147.29	33.04	69.36	68.60	-0.16	0.28	0.32	1.15	1.91
-24.00	100.00	180.00	313.20	32.4	5.00	71.62	5.73	69.17	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	147.46	33.51	70.49	69.17	-0.16	0.28	0.32	0.59	1.91
-25.00	100.00	180.00	315.00	32.5	5.00	72.19	5.73	69.75	6.67	59.14	6.31	-57.16	8.44	-38.89	8.71	41.07	147.62	33.99	71.62	69.75	-0.16	0.28	0.32	0.03	1.91
-26.00	100.00	180.00	316.80	32.6	5.00	72.20	5.73	70.32	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.81	147.77	34.46	72.19	70.32	-0.16	0.28	0.32	0.04	1.91
-27.00	100.00	180.00	318.60	32.7	5.00	72.22	5.73	70.89	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	147.93	34.94	72.20	70.89	-0.16	0.28	0.32	0.61	1.91
-28.00	100.00	180.00	320.40	32.8	5.00	72.80	5.73	71.46	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	148.08	35.41	72.22	71.46	-0.17	0.28	0.33	1.16	1.91
-29.00	100.00	180.00	322.20	32.9	5.00	73.93	5.73	72.04	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	148.23	35.89	72.80	72.04	-0.17	0.28	0.33	1.15	1.91
-30.00	100.00	180.00	324.00	33	5.00	75.07	5.73	72.61	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	148.36	36.37	73.93	72.61	-0.18	0.28	0.33	0.59	1.91
-31.00	100.00	180.00	325.80	33.1	5.00	75.63	5.73	73.18	6.67	59.14	6.31	-57.15	8.44	-38.88	8.71	41.07	148.49	36.86	75.07	73.18	-0.18	0.28	0.33	0.03	1.91
-32.00	100.00	180.00	327.60	33.2	5.00	75.64	5.73	73.76	6.63	58.91	6.35	-57.38	8.47	-39.14	8.68	40.82	148.62	37.34	75.63	73.76	-0.17	0.28	0.33	0.04	1.91
-33.00	100.00	180.00	329.40	33.3	5.00	75.66	5.73	74.33	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	148.74	37.82	75.64	74.33	-0.18	0.28	0.33	0.60	1.91
-34.00	100.00	180.00	331.20	33.4	5.00	76.24	5.73	74.90	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	148.86	38.31	75.66	74.90	-0.18	0.29	0.34	1.16	1.91
-35.00	100.00	180.00	333.00	33.5	5.00	77.38	5.73	75.48	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	148.98	38.79	76.24	75.48	-0.19	0.29	0.35	1.15	1.91
-36.00	100.00	180.00	334.80	33.6	5.00	78.51	5.73	76.05	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	149.09	39.28	77.38	76.05	-0.19	0.29	0.35	0.59	1.91
-37.00	100.00	180.00	336.60	33.7	5.00	79.08	5.73	76.62	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	149.19	39.77	78.51	76.62	-0.19	0.29	0.35	0.03	1.91
-38.00	100.00	180.00	338.40	33.8	5.00	79.08	5.73	77.20	6.63	58.91	6.35	-57.38	8.47	-39.13	8.68	40.82	149.29	40.26	79.08	77.20	-0.19	0.29	0.34	0.03	1.91
-39.00	100.00	180.00	340.20	33.9	5.00	79.10	5.73	77.77	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	149.38	40.75	79.08	77.77	-0.19	0.29	0.35	0.60	1.91
-40.00	100.00	180.00	342.00	34	5.00	79.67	5.73	78.34	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	149.48	41.25	79.10	78.34	-0.20	0.29	0.35	1.16	1.91
-41.00	100.00	180.00	343.80	34.1	5.00	80.82	5.73	78.92	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	149.57	41.74	79.67	78.92	-0.21	0.29	0.36	1.15	1.91
-42.00	100.00	180.00	345.60	34.2	5.00	81.95	5.73	79.49	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.08	149.65	42.23	80.82	79.49	-0.21	0.29	0.36	0.59	1.91
-43.00	100.00	180.00	347.40	34.3	5.00	82.52	5.73	80.06	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.07	149.72	42.73	81.95	80.06	-0.21	0.29	0.36	0.02	1.91
-44.00	100.00	180.00	349.20	34.4	5.00	82.53	5.73	80.64	6.63	58.91	6.35	-57.38	8.47	-39.13	8.68	40.82	149.78	43.22	82.52	80.64	-0.20	0.29	0.36	0.03	1.91
-45.00	100.00	180.00	351.00	34.5	5.00	82.54	5.73	81.21	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	149.85	43.72	82.53	81.21	-0.21	0.29	0.36	0.59	1.91
-46.00	100.00	180.00	352.80	34.6	5.00	83.11	5.73	81.78	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	149.91	44.21	82.54	81.78	-0.21	0.29	0.36	1.16	1.91
-47.00	100.00	180.00	354.60	34.7	5.00	84.26	5.73	82.36	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	149.97	44.71	83.11	82.36	-0.22	0.30	0.37	1.15	1.91
-48.00	100.00	180.00	356.40	34.8	5.00	85.40	5.73	82.93	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.08	150.02	45.21	84.26	82.93	-0.22	0.30	0.37	0.59	1.91
-49.00	100.00	180.00	358.20	34.9	5.00	85.97	5.73	83.50	6.67	59.15	6.31	-57.15	8.44	-38.88	8.72	41.08	150.06	45.71	85.40	83.50	-0.22	0.30	0.37	0.02	1.91
-50.00	100.00	180.00	360.00	35	5.00	85.97	5.73	84.08	6.63	58.92	6.35	-57.38	8.47	-39.13	8.68	40.82	150.10	46.20	85.97	84.08	-0.22	0.30	0.37	0.02	1.91
-51.00	100.00	180.00	0.00	35.1	5.00	85.98	5.73	84.65	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	150.13	46.70	85.97	84.65	-0.22	0.30	0.37	0.59	1.91
-52.00	100.00	180.00	0.00	35.2	5.00	86.55	5.73	85.22	6.59	58.68	6.39	-57.60	8.50	-39.38	8.65	40.57	150.17	47.20	85.98	85.22	-0.23	0.30	0.38	1.16	1.91
-53.00	100.00	180.00	0.00	35.3	5.00	87.70	5.73	85.80	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	150.20	47.70	86.55	85.80	-0.24	0.30	0.38	1.16	1.91
-54.00	100.00	180.00	0.00	35.4	5.00	88.84	5.73	86.37	6.67	59.15	6.30	-57.14	8.44	-38.87	8.72	41.08	150.22	48.20	87.70	86.37	-0.24	0.30	0.38	0.58	1.91

-55.00	100.00	180.00	0.00	35.5	5.00	89.41	5.73	86.94	6.67	59.15	6.30	-57.15	8.44	-38.87	8.72	41.08	150.23	48.70	88.84	86.94	-0.24	0.30	0.38	0.01	1.91
-56.00	100.00	180.00	0.00	35.6	5.00	89.41	5.73	87.52	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	150.23	49.20	89.41	87.52	-0.23	0.30	0.38	0.01	1.91
-57.00	100.00	180.00	0.00	35.7	5.00	89.42	5.73	88.09	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	150.24	49.70	89.41	88.09	-0.24	0.30	0.38	0.59	1.91
-58.00	100.00	180.00	0.00	35.8	5.00	89.99	5.73	88.66	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	150.24	50.20	89.42	88.66	-0.25	0.30	0.39	1.16	1.91
-59.00	100.00	180.00	0.00	35.9	5.00	91.14	5.73	89.24	6.63	58.92	6.35	-57.37	8.47	-39.13	8.68	40.82	150.24	50.70	89.99	89.24	-0.25	0.30	0.39	1.16	1.91
-60.00	100.00	180.00	0.00	36	5.00	92.28	5.73	89.81	6.67	59.16	6.30	-57.14	8.44	-38.87	8.72	41.08	150.23	51.20	91.14	89.81	-0.26	0.30	0.39	0.58	1.91
-61.00	100.00	180.00	0.00	36.1	5.00	92.86	5.73	90.38	6.67	59.16	6.30	-57.14	8.44	-38.87	8.72	41.08	150.21	51.70	92.28	90.38	-0.25	0.30	0.39	0.01	1.91
-62.00	100.00	180.00	0.00	36.2	5.00	92.86	5.73	90.96	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.83	150.19	52.20	92.86	90.96	-0.25	0.30	0.39	0.01	1.91
-63.00	100.00	180.00	0.00	36.3	5.00	92.86	5.73	91.53	6.59	58.68	6.39	-57.60	8.50	-39.38	8.65	40.57	150.16	52.70	92.86	91.53	-0.25	0.30	0.39	0.58	1.91
-64.00	100.00	180.00	0.00	36.4	5.00	93.43	5.73	92.10	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	150.14	53.20	92.86	92.10	-0.26	0.30	0.40	1.16	1.91
-65.00	100.00	180.00	0.00	36.5	5.00	94.58	5.73	92.68	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.82	150.11	53.70	93.43	92.68	-0.27	0.30	0.40	1.16	1.91
-66.00	100.00	180.00	0.00	36.6	5.00	95.73	5.73	93.25	6.67	59.16	6.30	-57.14	8.44	-38.87	8.72	41.08	150.07	54.20	94.58	93.25	-0.27	0.30	0.40	0.58	1.91
-67.00	100.00	180.00	0.00	36.7	5.00	96.31	5.73	93.82	6.67	59.16	6.30	-57.14	8.44	-38.87	8.72	41.09	150.02	54.69	95.73	93.82	-0.27	0.30	0.40	0.00	1.91
-68.00	100.00	180.00	0.00	36.8	5.00	96.30	5.73	94.39	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	149.96	55.19	96.31	94.39	-0.27	0.30	0.40	0.00	1.91
-69.00	100.00	180.00	0.00	36.9	5.00	96.30	5.73	94.97	6.59	58.68	6.39	-57.60	8.50	-39.38	8.65	40.57	149.91	55.69	96.30	94.97	-0.27	0.30	0.40	0.58	1.91
-70.00	100.00	180.00	0.00	37	5.00	96.87	5.73	95.54	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	149.85	56.18	96.30	95.54	-0.28	0.30	0.41	1.16	1.91
-71.00	100.00	180.00	0.00	37.1	5.00	98.02	5.73	96.11	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.83	149.79	56.68	96.87	96.11	-0.28	0.30	0.41	1.16	1.91
-72.00	100.00	180.00	0.00	37.2	5.00	99.17	5.73	96.69	6.67	59.16	6.30	-57.14	8.44	-38.87	8.72	41.09	149.72	57.18	98.02	96.69	-0.29	0.30	0.41	0.58	1.91
-73.00	100.00	180.00	0.00	37.3	5.00	99.75	5.73	97.26	6.67	59.16	6.30	-57.14	8.44	-38.86	8.72	41.09	149.64	57.67	99.17	97.26	-0.28	0.30	0.41	0.00	1.91
-74.00	100.00	180.00	0.00	37.4	5.00	99.75	5.73	97.83	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	149.56	58.16	99.75	97.83	-0.28	0.30	0.41	-0.01	1.91
-75.00	100.00	180.00	0.00	37.5	5.00	99.74	5.73	98.41	6.59	58.68	6.39	-57.60	8.50	-39.38	8.65	40.57	149.48	58.65	99.75	98.41	-0.28	0.30	0.41	0.57	1.91
-76.00	100.00	180.00	0.00	37.6	5.00	100.30	5.73	98.98	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	149.39	59.15	99.74	98.98	-0.29	0.30	0.42	1.16	1.91
-77.00	100.00	180.00	0.00	37.7	5.00	101.46	5.73	99.55	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.83	149.30	59.64	100.30	99.55	-0.30	0.30	0.42	1.16	1.91
-78.00	100.00	180.00	0.00	37.8	5.00	102.62	5.73	100.13	6.67	59.16	6.30	-57.14	8.44	-38.86	8.72	41.09	149.20	60.13	101.46	100.13	-0.30	0.30	0.42	0.58	1.91
-79.00	100.00	180.00	0.00	37.9	5.00	103.20	5.73	100.70	6.67	59.17	6.30	-57.13	8.44	-38.86	8.72	41.09	149.09	60.62	102.62	100.70	-0.30	0.30	0.42	-0.01	1.91
-80.00	100.00	180.00	0.00	38	5.00	103.19	5.73	101.27	6.63	58.93	6.35	-57.37	8.47	-39.12	8.69	40.83	148.98	61.10	103.20	101.27	-0.30	0.30	0.42	-0.02	1.91
-81.00	100.00	180.00	0.00	38.1	5.00	103.18	5.73	101.85	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	148.86	61.59	103.19	101.85	-0.30	0.30	0.42	0.57	1.91
-82.00	100.00	180.00	0.00	38.2	5.00	103.74	5.73	102.42	6.59	58.68	6.39	-57.61	8.50	-39.38	8.65	40.57	148.75	62.08	103.18	102.42	-0.31	0.30	0.43	1.15	1.91
-83.00	100.00	180.00	0.00	38.3	5.00	104.90	5.73	102.99	6.63	58.92	6.35	-57.37	8.47	-39.13	8.69	40.83	148.63	62.56	103.74	102.99	-0.31	0.30	0.43	1.16	1.91
-84.00	100.00	180.00	0.00	38.4	5.00	106.06	5.73	103.57	6.67	59.16	6.30	-57.13	8.44	-38.86	8.72	41.09	148.50	63.05	104.90	103.57	-0.32	0.30	0.43	0.58	1.91
-85.00	100.00	180.00	0.00	38.5	5.00	106.65	5.73	104.14	6.67	59.17	6.30	-57.13	8.44	-38.86	8.72	41.10	148.36	63.53	106.06	104.14	-0.31	0.30	0.43	-0.01	1.91
-86.00	100.00	180.00	0.00	38.6	5.00	106.64	5.73	104.71	6.63	58.93	6.35	-57.36	8.47	-39.12	8.69	40.84	148.22	64.01	106.65	104.71	-0.31	0.30	0.43	-0.02	1.91
-87.00	100.00	180.00	0.00	38.7	5.00	106.62	5.73	105.29	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	148.08	64.49	106.64	105.29	-0.31	0.30	0.43	0.56	1.91
-88.00	100.00	180.00	0.00	38.8	5.00	107.18	5.73	105.86	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	147.94	64.96	106.62	105.86	-0.32	0.30	0.44	1.15	1.91
-89.00	100.00	180.00	0.00	38.9	5.00	108.34	5.73	106.43	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	147.79	65.44	107.18	106.43	-0.33	0.29	0.44	1.16	1.91
-90.00	100.00	180.00	0.00	39	5.00	109.51	5.73	107.01	6.67	59.17	6.30	-57.13	8.44	-38.86	8.72	41.09	147.63	65.92	108.34	107.01	-0.33	0.29	0.44	0.58	1.91
-91.00	100.00	180.00	0.00	39.1	5.00	110.10	5.73	107.58	6.67	59.17	6.30	-57.13	8.44	-38.85	8.72	41.10	147.46	66.39	109.51	107.58	-0.33	0.29	0.44	-0.02	1.91
-92.00	100.00	180.00	0.00	39.2	5.00	110.08	5.73	108.15	6.63	58.93	6.35	-57.36	8.47	-39.11	8.69	40.84	147.29	66.86	110.10	108.15	-0.33	0.30	0.44	-0.03	1.91
-93.00	100.00	180.00	0.00	39.3	5.00	110.06	5.73	108.73	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	147.12	67.33	110.08	108.73	-0.33	0.30	0.44	0.55	1.91
-94.00	100.00	180.00	0.00	39.4	5.00	110.62	5.73	109.30	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	146.95	67.80	110.06	109.30	-0.34	0.29	0.45	1.15	1.91
-95.00	100.00	180.00	0.00	39.5	5.00	111.78	5.73	109.87	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	146.77	68.26	110.62	109.87	-0.34	0.29	0.45	1.17	1.91
-96.00	100.00	180.00	0.00	39.6	5.00	112.95	5.73	110.45	6.67	59.17	6.30	-57.13	8.44	-38.86	8.72	41.10	146.59	68.73	111.78	110.45	-0.35	0.29	0.45	0.58	1.91
-97.00	100.00	180.00	0.00	39.7	5.00	113.54	5.73	111.02	6.68	59.18	6.30	-57.12	8.44	-38.85	8.72	41.10	146.39	69.19	112.95	111.02	-0.34	0.29	0.45	-0.02	1.91
-98.00	100.00	180.00	0.00	39.8	5.00	113.53	5.73	111.59	6.63	58.93	6.34	-57.36	8.47	-39.11	8.69	40.84	146.19	69.65	113.54	111.59	-0.34	0.29	0.45	-0.04	1.91
-99.00	100.00	180.00	0.00	39.9	5.00	113.50	5.73	112.17	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	145.99	70.11	113.53	112.17	-0.34	0.29	0.45	0.55	1.91
-100.00	100.00	180.00	0.00	40	5.00	114.06	5.73	112.74	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	145.79	70.56	113.50	112.74	-0.35	0.29	0.45	1.15	1.91
-101.00	99.99	181.15	0.00	40.1	5.00	115.22	5.73	113.31	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	145.59	71.02	114.06	113.31	-0.36	0.29	0.46	1.17	1.91
-102.00	99.96	182.29	0.00	40.2	5.00	116.40	5.73	113.89	6.67	59.17	6.30	-57.13	8.44	-38.85	8.72	41.10	145.38	71.47	115.22	113.89	-0.36	0.28	0.46	0.58	1.91
-103.00	99.91	183.44	0.00	40.3	5.00	116.99	5.73	114.46	6.68	59.18	6.30	-57.12	8.44	-38.84	8.72	41.11	145.15	71.92	116.40	114.46	-0.36	0.29	0.46	-0.03	1.91
-104.00	99.84	184.59	0.00	40.4	5.00	116.98	5.73	115.03	6.63	58.94	6.34	-57.36	8.47	-39.11	8.69	40.84	144.93	72.37	116.99	115.03	-0.35	0.29	0.46	-0.05	1.91
-105.00	99.75	185.74	0.00	40.5	5.00	116.94	5.73	115.61	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	144.70	72.81	116.98	115.61	-0.36	0.29	0.46	0.54	1.91
-106.00	99.64	186.89	0.00	40.6	5.00	117.49	5.73	116.18	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	144.47	73.26	116.94	116.18	-0.36	0.28	0.46	1.15	1.91
-107.00	99.51	188.05	0.00	40.7	5.00	118.66	5.73	116.75	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	144.24	73.70	117.49	116.75	-0.37	0.28	0.46	1.17	1.91
-108.00	99.36	189.21	0.00	40.8	5.00	119.85	5.73	117.32	6.67	59.17	6.30	-57.12	8.44	-38.85	8.72	41.10	144.00	74.14	118.66	117.32	-0.37	0.28	0.47	0.58	1.91
-109.00	99.18	190.37	0.00	40.9	5.00	120.44	5.73	117.90	6.68	59.18	6.30	-57.12	8.43	-38.84	8.72	41.11	143.75	74.57	119.85	117.90	-0.37	0.28	0.46	-0.04	1.91
-110.00	98.99	191.54	0.00	41	5.00	120.42	5.73	118.47	6.63	58.94	6.34	-57.35	8.47	-39.10	8.69	40.85	143.50	75.01	120.44	118.47	-0.37	0.28	0.46	-0.06	1.91
-111.00	98.77	192.71	0.00	41.1	5.00	120.38	5.73	119.04	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	143.25	75.44	120.42	119.04	-0.37	0.28	0.46	0.53	1.91
-112.00	98.54	193.89	0.00	41.2	5.00	120.93	5.73	119.62	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	142.99	75.87	120.38	119.62	-0.38	0.28	0.47	1.15	1.91
-113.00	98.28	195.07	0.00	41.3	5.00	122.10	5.73	120.19	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	142.74	76.30	120.93	120.19	-0.38	0.27	0.47	1.17	1.91
-114.00	98.00	196.26	0.00	41.4	5.00	123.29	5.73	120.76	6.68	59.18	6.30	-57.12	8.44	-38.85	8.72	41.11	142.47	76.72	122.10	120.76	-0.38	0.27	0.47	0.58	1.91
-115.00	97.70	197.46	0.00	41.5	5.00	123.89	5.73	121.34	6.68	59.19	6.30	-57.11	8.43	-38.84	8.72	41.12	142.20	77.14	123.29	121.34	-0.38	0.27	0.47	-0.04	1.91
-116.00	97.37	198.66	0.00	41.6	5.00	123.87	5.73	121.91	6.63	58.94	6.34	-57.35	8.47	-39.10	8.69	40.85	141.92	77.55	123.89	121.91	-0.38	0.28	0.47	-0.07	1.91
-117.00	97.02	199.88	0.00	41.7	5.00	123.82	5.73	122.48	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	141.64	77.97	123.87	122.48	-0.38	0.28	0.47	0.53	1.91
-118.00	96.65	201.10	0.00	41.8	5.00	124.37	5.73	123.06	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	141.36	78.38	123.82	123.06	-0.39	0.27	0.47	1.15	1.91
-119.00	96.25	202.33	0.00	41.9	5.00	125.54	5.73	123.63	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	141.08	78.80	124.37	123.63	-0.39	0.27	0.48	1.17	1.91
-120.00	95.83	203.58	0.00	42	5.00	126.74	5.73	124.20	6.68	59.18	6.30	-57.12	8.44	-38.84	8.72	41.11	140.79	79.20	125.54	124.20	-0.40	0.27	0.48	0.58	1.91
-121.00	95.38	204.83	0.00	42.1	5.00	127.34	5.73	124.78	6.68	59.19	6.30	-57.11	8.43	-38.83	8.72	41.12	140.49	79.60	126.74	124.78	-0.39	0.27	0.48	-0.05	1.91
-122.00	94.90	206.10	0.00	42.2	5.00	127.31	5.73	125.35	6.63	58.95	6.34	-57.35	8.47	-39.10	8.69	40.85	140.19	80.00	127.34	125.35	-0.39	0.27	0.48	-0.08	1.91
-123.00	94.40	207.39	0.00	42.3	5.00	127.26	5.73	125.92	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	139.88	80.40	127.31	125.92	-0.39	0.27	0.48	0.52	1.91
-124.00	93.86	208.69	0.00	42.4	5.00	127.80	5.73	126.50	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	139.58	80.80	127.26	126.50	-0.40	0.27	0.48	1.15	1.91
-125.00	93.30	210.00	0.00	42.5	5.00	128.98	5.73	127.07	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	139.27	81.19	127.80	127.07	-0.41	0.26	0.48	1.18	1.91
-126.00	92.71	211.33	0.00	42.6	5.00	130.18	5.73	127.64	6.68	59.18	6.30	-57.12	8.44	-38.84	8.72	41.11	138.96	81.58	128.98	127.64	-0.41	0.26	0.48	0.57	1.91
-127.00	92.08	212.68	0.00	42.7	5.00	130.79	5.73	128.22	6.68	59.20	6.30	-57.10	8.43	-38.83	8.72	41.13	138.64	81.96	130.18	128.22	-0.41	0.26	0.48	-0.06	1.91
-128.00	91.42	214.06	0.00	42.8	5.00	130.76	5.73	128.79	6.63	58.95	6.34	-57.35	8.47	-39.10	8.69	40.86	138.31	82.34	130.79	128.79	-0.40	0.26	0.48	-0.09	1.91
-129.00	90.73	215.45	0.00	42.9	5.00	130.70	5.73	129.36	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	137.98	82.72	130.76	129.36	-0.41	0.26	0.48	0.51	1.91
-130.00	90.00	216.87	0.00	43	5.00	131.24	5.73	129.94	6.59	58.68	6.39	-57.61	8.51	-39.39	8.65	40.56	137.66	83.10	130.70	129.94	-0.41	0.26	0.49	1.15	1.91
-131.00	89.23	218.32	0.00	43.1	5.00	132.42	5.73	130.51	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	137.33	83.48	131.24	130.51	-0.42	0.25	0.49	1.18	1.91
-132.00	88.42	219.79	0.00	43.2	5.00	133.63	5.73	131.08	6.68	59.19	6.30	-57.11	8.43	-38.84	8.72	41.11	136.99	83.84	132.42	131.08	-0.42	0.25	0.49	0.57	1.91
-133.00	87.56	221.30	0.00	43.3	5.00	134.24	5.73	131.66	6.68	59.20	6.30	-57.10	8.43	-38.82	8.72	41.13	136.65	84.21	133.63	131.66	-0.42	0.25	0.49	-0.06	1.91
-134.00	86.66	222.84	0.00	43.4	5.00	134.21	5.73	132.23	6.64	58.95	6.34	-57.34	8.47	-39.09	8.69	40.86	136.30	84.56	134.24	132.23	-0.42	0.26	0.49	-0.10	1.91
-135.00	85.71	224.43	0.00	43.5	5.00	134.14	5.73	132.80	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	135.95	84.92	134.21	132.80	-0.42	0.25	0.49	0.51	1.91
-136.00	84.70	226.05	0.00	43.6	5.00	134.68	5.73	133.38	6.59	58.67	6.39	-57.61	8.51	-39.39	8.65	40.56	135.60	85.28	134.14	133.38	-0.42	0.25	0.49	1.15	1.91
-137.00	83.63	227.73	0.00	43.7	5.00	135.86	5.73	133.95	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	135.25	85.64	134.68	133.95	-0.43	0.24	0.49	1.18	1.91
-138.00	82.50	229.46	0.00	43.8	5.00	137.08	5.73	134.52	6.68	59.19	6.30	-57.11	8.43	-38.84	8.72	41.12	134.89	85.99	135.86	134.52	-0.43	0.24	0.49	0.57	1.91
-139.00	81.29	231.26	0.00	43.9	5.00	137.69	5.73	135.10	6.68	59.20	6.30	-57.10	8.43	-38.82	8.72	41.13	134.52	86.33	137.08	135.10	-0.43	0.25	0.49	-0.07	1.91
-140.00	80.00	233.13	0.00	44	5.00	137.65	5.73	135.67	6.64	58.95	6.34	-57.34	8.47	-39.09	8.69	40.86	134.15	86.66	137.69	135.67	-0.43	0.25	0.49	-0.11	1.91
-141.00	78.62	235.08	0.00	44.1	5.00	137.58	5.73	136.24	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	133.78	87.00	137.65	136.24	-0.43	0.25	0.49	0.50	1.91
-142.00	77.13	237.14	0.00	44.2	5.00	138.12	5.73	136.82	6.59	58.67	6.39	-57.62	8.51	-39.39	8.65	40.56	133.42	87.34	137.58	136.82	-0.43	0.24	0.50	1.14	1.91
-143.00	75.51	239.32	0.00	44.3	5.00	139.30	5.73	137.39	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	133.04	87.67	138.12	137.39	-0.44	0.24	0.50	1.18	1.91
-144.00	73.75	241.64	0.00	44.4	5.00	140.52	5.73	137.96	6.68	59.19	6.30	-57.11	8.43	-38.83	8.72	41.12	132.66	88.00	139.30	137.96	-0.44	0.23	0.50	0.57	1.91
-145.00	71.79	244.16	0.00	44.5	5.00	141.14	5.73	138.54	6.68	59.21	6.29	-57.09	8.43	-38.81	8.72	41.14	132.28	88.31	140.52	138.54	-0.44	0.24	0.50	-0.08	1.91
-146.00	69.60	246.93	0.00	44.6	5.00	141.10	5.73	139.11	6.64	58.96	6.34	-57.34	8.47	-39.09	8.69	40.87	131.89	88.63	141.14	139.11	-0.44	0.24	0.50	-0.12	1.91
-147.00	67.06	250.05	0.00	44.7	5.00	141.02	5.73	139.68	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	131.50	88.94	141.10	139.68	-0.44	0.24	0.50	0.49	1.91
-148.00	64.00	253.74	0.00	44.8	5.00	141.55	5.73	140.25	6.59	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	131.11	89.26	141.02	140.25	-0.44	0.23	0.50	1.14	1.91
-149.00	59.95	258.52	0.00	44.9	5.00	142.74	5.73	140.83	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	130.72	89.57	141.55	140.83	-0.45	0.23	0.50	1.19	1.91
-150.00	50.00	270.00	0.00	45	5.00	143.97	5.73	141.40	6.68	59.19	6.30	-57.11	8.43	-38.83	8.72	41.12	130.32	89.87	142.74	141.40	-0.45	0.22	0.50	0.57	1.91
-149.00	40.05	281.48	0.00	45.1	5.00	144.59	5.73	141.97	6.68	59.21	6.29	-57.09	8.43	-38.81	8.72	41.14	129.92	90.16	143.97	141.97	-0.45	0.23	0.50	-0.08	1.91
-148.00	36.00	286.26	0.00	45.2	5.00	144.55	5.73	142.55	6.64	58.96	6.34	-57.33	8.47	-39.08	8.69	40.87	129.51	90.45	144.59	142.55	-0.45	0.23	0.50	-0.13	1.91
-147.00	32.94	289.95	0.00	45.3	5.00	144.46	5.73	143.12	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	129.10	90.74	144.55	143.12	-0.45	0.23	0.50	0.48	1.91
-146.00	30.40	293.07	0.00	45.4	5.00	144.99	5.73	143.69	6.59	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	128.70	91.04	144.46	143.69	-0.45	0.22	0.50	1.14	1.91
-145.00	28.21	295.84	0.00	45.5	5.00	146.18	5.73	144.27	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	128.29	91.32	144.99	144.27	-0.46	0.22	0.50	1.19	1.91
-144.00	26.25	298.36	0.00	45.6	5.00	147.41	5.73	144.84	6.68	59.20	6.30	-57.10	8.43	-38.83	8.72	41.13	127.87	91.60	146.18	144.84	-0.46	0.21	0.51	0.57	1.91
-143.00	24.49	300.68	0.00	45.7	5.00	148.04	5.73	145.41	6.68	59.22	6.29	-57.08	8.43	-38.81	8.72	41.15	127.45	91.87	147.41	145.41	-0.46	0.22	0.51	-0.09	1.91
-142.00	22.87	302.86	0.00	45.8	5.00	147.99	5.73	145.99	6.64	58.96	6.34	-57.33	8.47	-39.08	8.69	40.87	127.03	92.13	148.04	145.99	-0.45	0.22	0.50	-0.14	1.91
-141.00	21.38	304.92	0.00	45.9	5.00	147.90	5.73	146.56	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	126.60	92.40	147.99	146.56	-0.45	0.22	0.51	0.48	1.91
-140.00	20.00	306.87	0.00	46	5.00	148.43	5.73	147.13	6.58	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	126.18	92.66	147.90	147.13	-0.46	0.21	0.51	1.14	1.91
-139.00	18.71	308.74	0.00	46.1	5.00	149.62	5.73	147.71	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	125.75	92.93	148.43	147.71	-0.46	0.21	0.51	1.19	1.91
-138.00	17.50	310.54	0.00	46.2	5.00	150.86	5.73	148.28	6.68	59.20	6.30	-57.10	8.43	-38.82	8.72	41.13	125.32	93.18	149.62	148.28	-0.46	0.20	0.51	0.57	1.91
-137.00	16.37	312.27	0.00	46.3	5.00	151.49	5.73	148.85	6.68	59.22	6.29	-57.08	8.43	-38.80	8.72	41.15	124.88	93.42	150.86	148.85	-0.46	0.21	0.51	-0.10	1.91
-136.00	15.30	313.95	0.00	46.4	5.00	151.44	5.73	149.43	6.64	58.97	6.34	-57.33	8.47	-39.08	8.69	40.87	124.44	93.66	151.49	149.43	-0.46	0.21	0.51	-0.15	1.91
-135.00	14.29	315.57	0.00	46.5	5.00	151.34	5.73	150.00	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	124.01	93.90	151.44	150.00	-0.46	0.21	0.51	0.47	1.91
-134.00	13.34	317.16	0.00	46.6	5.00	151.86	5.73	150.57	6.58	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	123.57	94.14	151.34	150.57	-0.47	0.20	0.51	1.14	1.91
-133.00	12.44	318.70	0.00	46.7	5.00	153.06	5.73	151.15	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	123.13	94.38	151.86	151.15	-0.47	0.20	0.51	1.19	1.91
-132.00	11.58	320.21	0.00	46.8	5.00	154.31	5.73	151.72	6.68	59.20	6.30	-57.10	8.43	-38.82	8.72	41.13	122.68	94.60	153.06	151.72	-0.47	0.19	0.51	0.57	1.91
-131.00	10.77	321.68	0.00	46.9	5.00	154.93	5.73	152.29	6.68	59.22	6.29	-57.08	8.43	-38.80	8.73	41.16	122.23	94.82	154.31	152.29	-0.47	0.20	0.51	-0.10	1.91
-130.00	10.00	323.13	0.00	47	5.00	154.89	5.73	152.87	6.64	58.97	6.34	-57.33	8.47	-39.07	8.69	40.88	121.78	95.03	154.93	152.87	-0.47	0.20	0.51	-0.16	1.91
-129.00	9.27	324.55	0.00	47.1	5.00	154.78	5.73	153.44	6.59	58.69	6.39	-57.60	8.50	-39.38	8.65	40.57	121.32	95.24	154.89	153.44	-0.47	0.20	0.51	0.46	1.91
-128.00	8.58	325.94	0.00	47.2	5.00	155.30	5.73	154.01	6.58	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	120.87	95.46	154.78	154.01	-0.47	0.19	0.51	1.14	1.91
-127.00	7.92	327.32	0.00	47.3	5.00	156.50	5.73	154.59	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	120.42	95.67	155.30	154.59	-0.48	0.19	0.51	1.20	1.91

-126.00	7.29	328.67	0.00	47.4	5.00	157.75	5.73	155.16	6.68	59.20	6.30	-57.10	8.43	-38.82	8.72	41.13	119.96	95.86	156.50	155.16	-0.48	0.18	0.51	0.57	1.91
-125.00	6.70	330.00	0.00	47.5	5.00	158.38	5.73	155.73	6.68	59.23	6.29	-57.07	8.43	-38.79	8.73	41.16	119.50	96.05	157.75	155.73	-0.48	0.19	0.51	-0.11	1.91
-124.00	6.14	331.31	0.00	47.6	5.00	158.33	5.73	156.31	6.64	58.97	6.34	-57.32	8.46	-39.07	8.69	40.88	119.03	96.24	158.38	156.31	-0.48	0.19	0.51	-0.17	1.91
-123.00	5.60	332.61	0.00	47.7	5.00	158.22	5.73	156.88	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	118.57	96.42	158.33	156.88	-0.48	0.19	0.51	0.46	1.91
-122.00	5.10	333.90	0.00	47.8	5.00	158.74	5.73	157.45	6.58	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	118.10	96.61	158.22	157.45	-0.48	0.18	0.51	1.14	1.91
-121.00	4.62	335.17	0.00	47.9	5.00	159.94	5.73	158.03	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	117.64	96.79	158.74	158.03	-0.48	0.18	0.51	1.20	1.91
-120.00	4.17	336.42	0.00	48	5.00	161.20	5.73	158.60	6.68	59.21	6.30	-57.09	8.43	-38.82	8.72	41.14	117.17	96.96	159.94	158.60	-0.48	0.17	0.51	0.57	1.91
-119.00	3.75	337.67	0.00	48.1	5.00	161.83	5.73	159.17	6.68	59.23	6.29	-57.07	8.43	-38.79	8.73	41.16	116.69	97.12	161.20	159.17	-0.48	0.18	0.51	-0.11	1.91
-118.00	3.35	338.90	0.00	48.2	5.00	161.78	5.73	159.75	6.64	58.97	6.34	-57.32	8.46	-39.07	8.69	40.88	116.22	97.28	161.83	159.75	-0.48	0.18	0.51	-0.18	1.91
-117.00	2.98	340.12	0.00	48.3	5.00	161.66	5.73	160.32	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	115.74	97.43	161.78	160.32	-0.48	0.18	0.51	0.45	1.91
-116.00	2.63	341.34	0.00	48.4	5.00	162.17	5.73	160.89	6.58	58.67	6.39	-57.62	8.51	-39.40	8.65	40.55	115.27	97.59	161.66	160.89	-0.48	0.17	0.51	1.14	1.91
-115.00	2.30	342.54	0.00	48.5	5.00	163.38	5.73	161.46	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	114.79	97.75	162.17	161.46	-0.49	0.16	0.51	1.20	1.91
-114.00	2.00	343.74	0.00	48.6	5.00	164.64	5.73	162.04	6.68	59.21	6.29	-57.09	8.43	-38.81	8.72	41.14	114.31	97.89	163.38	162.04	-0.49	0.16	0.51	0.57	1.91
-113.00	1.72	344.93	0.00	48.7	5.00	165.28	5.73	162.61	6.69	59.24	6.29	-57.07	8.43	-38.78	8.73	41.17	113.83	98.02	164.64	162.61	-0.49	0.17	0.52	-0.12	1.91
-112.00	1.46	346.11	0.00	48.8	5.00	165.23	5.73	163.18	6.64	58.98	6.34	-57.32	8.46	-39.06	8.69	40.89	113.35	98.15	165.28	163.18	-0.49	0.17	0.52	-0.18	1.91
-111.00	1.23	347.29	0.00	48.9	5.00	165.10	5.73	163.76	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	112.86	98.28	165.23	163.76	-0.49	0.17	0.52	0.44	1.91
-110.00	1.01	348.46	0.00	49	5.00	165.61	5.73	164.33	6.58	58.66	6.39	-57.62	8.51	-39.40	8.65	40.55	112.38	98.40	165.10	164.33	-0.49	0.16	0.52	1.14	1.91
-109.00	0.82	349.63	0.00	49.1	5.00	166.82	5.73	164.90	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	111.90	98.53	165.61	164.90	-0.49	0.15	0.52	1.20	1.91
-108.00	0.64	350.79	0.00	49.2	5.00	168.09	5.73	165.48	6.68	59.21	6.29	-57.09	8.43	-38.81	8.72	41.14	111.41	98.64	166.82	165.48	-0.49	0.15	0.52	0.57	1.91
-107.00	0.49	351.95	0.00	49.3	5.00	168.73	5.73	166.05	6.69	59.24	6.29	-57.06	8.43	-38.78	8.73	41.17	110.92	98.75	168.09	166.05	-0.49	0.16	0.52	-0.13	1.91
-106.00	0.36	353.11	0.00	49.4	5.00	168.67	5.73	166.62	6.64	58.98	6.34	-57.32	8.46	-39.06	8.69	40.89	110.43	98.84	168.73	166.62	-0.49	0.16	0.52	-0.19	1.91
-105.00	0.25	354.26	0.00	49.5	5.00	168.54	5.73	167.20	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	109.94	98.94	168.67	167.20	-0.49	0.16	0.52	0.44	1.91
-104.00	0.16	355.41	0.00	49.6	5.00	169.05	5.73	167.77	6.58	58.66	6.39	-57.63	8.51	-39.40	8.65	40.55	109.45	99.04	168.54	167.77	-0.49	0.15	0.52	1.14	1.91
-103.00	0.09	356.56	0.00	49.7	5.00	170.25	5.73	168.34	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	108.96	99.14	169.05	168.34	-0.50	0.14	0.52	1.21	1.91
-102.00	0.04	357.71	0.00	49.8	5.00	171.53	5.73	168.92	6.68	59.21	6.29	-57.09	8.43	-38.81	8.72	41.14	108.47	99.22	170.25	168.92	-0.50	0.14	0.52	0.57	1.91
-101.00	0.01	358.85	0.00	49.9	5.00	172.18	5.73	169.49	6.69	59.24	6.29	-57.06	8.43	-38.78	8.73	41.18	107.97	99.29	171.53	169.49	-0.50	0.14	0.52	-0.13	1.91
-100.00	0.00	360.00	0.00	50	5.00	172.12	5.73	170.06	6.64	58.98	6.34	-57.31	8.46	-39.06	8.69	40.89	107.48	99.36	172.18	170.06	-0.50	0.15	0.52	-0.20	1.91
-99.00	0.00	360.00	0.00	50.1	5.00	171.98	5.73	170.64	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	106.98	99.43	172.12	170.64	-0.50	0.15	0.52	0.43	1.91
-98.00	0.00	360.00	0.00	50.2	5.00	172.48	5.73	171.21	6.58	58.66	6.39	-57.63	8.51	-39.41	8.65	40.54	106.49	99.50	171.98	171.21	-0.50	0.14	0.52	1.14	1.91
-97.00	0.00	360.00	0.00	50.3	5.00	173.69	5.73	171.78	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	105.99	99.57	172.48	171.78	-0.50	0.13	0.52	1.21	1.91
-96.00	0.00	360.00	0.00	50.4	5.00	174.98	5.73	172.36	6.68	59.22	6.29	-57.09	8.43	-38.81	8.72	41.15	105.49	99.62	173.69	172.36	-0.50	0.13	0.52	0.57	1.91
-95.00	0.00	360.00	0.00	50.5	5.00	175.62	5.73	172.93	6.69	59.25	6.29	-57.06	8.43	-38.77	8.73	41.18	104.99	99.66	174.98	172.93	-0.50	0.13	0.52	-0.14	1.91
-94.00	0.00	360.00	0.00	50.6	5.00	175.56	5.73	173.50	6.64	58.98	6.34	-57.31	8.46	-39.06	8.69	40.90	104.50	99.70	175.62	173.50	-0.50	0.14	0.52	-0.21	1.91
-93.00	0.00	360.00	0.00	50.7	5.00	175.42	5.73	174.08	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	104.00	99.74	175.56	174.08	-0.50	0.14	0.52	0.43	1.91
-92.00	0.00	360.00	0.00	50.8	5.00	175.92	5.73	174.65	6.58	58.66	6.39	-57.63	8.51	-39.41	8.65	40.54	103.50	99.78	175.42	174.65	-0.50	0.13	0.52	1.14	1.91
-91.00	0.00	360.00	0.00	50.9	5.00	177.13	5.73	175.22	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	103.00	99.82	175.92	175.22	-0.50	0.12	0.51	1.21	1.91
-90.00	0.00	360.00	0.00	51	5.00	178.42	5.73	175.80	6.68	59.22	6.29	-57.08	8.43	-38.80	8.72	41.15	102.50	99.84	177.13	175.80	-0.50	0.12	0.51	0.57	1.91
-89.00	0.00	360.00	0.00	51.1	5.00	179.07	5.73	176.37	6.69	59.25	6.29	-57.05	8.43	-38.77	8.73	41.18	102.00	99.86	178.42	176.37	-0.50	0.12	0.52	-0.14	1.91
-88.00	0.00	360.00	0.00	51.2	5.00	179.01	5.73	176.94	6.64	58.99	6.34	-57.31	8.46	-39.05	8.69	40.90	101.50	99.86	179.07	176.94	-0.50	0.13	0.52	-0.22	1.91
-87.00	0.00	360.00	0.00	51.3	5.00	178.86	5.73	177.52	6.59	58.69	6.39	-57.60	8.50	-39.37	8.65	40.58	101.00	99.87	179.01	177.52	-0.50	0.13	0.52	0.42	1.91
-86.00	0.00	360.00	0.00	51.4	5.00	179.36	5.73	178.09	6.58	58.66	6.39	-57.63	8.51	-39.41	8.65	40.54	100.50	99.88	178.86	178.09	-0.50	0.12	0.52	1.14	1.91
-85.00	0.00	360.00	0.00	51.5	5.00	180.57	5.73	178.66	6.63	58.92	6.35	-57.37	8.47	-39.12	8.69	40.83	100.00	99.89	179.36	178.66	0.00	0.11	0.11	0.64	1.34
-84.00	0.00	360.00	0.00	51.6	5.00	181.19	4.01	179.24	5.60	45.97	5.36	-43.44	7.13	-31.16	7.31	33.45	99.50	99.88	180.57	179.24	0.00	0.12	0.12	-0.57	0.76
-83.00	0.00	360.00	0.00	51.7	5.00	180.60	2.29	179.64	4.93	27.64	4.86	-25.89	6.02	-20.65	6.08	22.12	99.00	99.87	181.19	179.64	0.00	0.13	0.13	-1.19	0.36
-82.00	0.00	360.00	0.00	51.8	5.00	179.38	1.09	179.87	4.81	12.07	4.83	-13.05	5.41	-11.62	5.39	10.74	98.50	99.87	180.60	179.87	0.00	0.13	0.13	-0.60	0.13
-81.00	0.00	360.00	0.00	51.9	5.00	178.76	0.40	179.98	4.90	3.27	4.91	-5.75	5.13	-5.50	5.12	3.13	98.00	99.87	179.38	179.98	0.00	0.13	0.13	0.62	0.02
-80.00	0.00	360.00	0.00	52	5.00	179.35	0.07	180.02	4.98	0.16	4.98	-1.49	5.02	-1.48	5.02	0.16	97.50	99.88	178.76	180.02	0.00	0.12	0.12	1.24	-0.02
-79.00	0.00	360.00	0.00	52.1	5.00	180.57	-0.05	180.02	5.01	0.04	5.01	1.05	4.99	1.06	4.99	0.04	97.00	99.89	179.35	180.02	0.00	0.11	0.11	0.65	-0.02
-78.00	0.00	360.00	0.00	52.2	5.00	181.20	-0.07	180.02	5.02	0.43	5.02	1.92	4.98	1.94	4.98	0.43	96.50	99.88	180.57	180.02	0.00	0.12	0.12	-0.57	-0.02
-77.00	0.00	360.00	0.00	52.3	5.00	180.60	-0.05	180.01	5.01	-0.01	5.02	1.19	4.99	1.19	4.98	-0.01	96.00	99.87	181.20	180.01	0.00	0.13	0.13	-1.20	-0.01
-76.00	0.00	360.00	0.00	52.4	5.00	179.38	-0.03	180.01	5.01	-1.00	5.01	-0.25	4.99	-0.25	4.99	-1.00	95.50	99.87	180.60	180.01	0.00	0.13	0.13	-0.60	-0.01
-75.00	0.00	360.00	0.00	52.5	5.00	178.75	-0.02	180.00	5.01	-1.44	5.00	-1.05	4.99	-1.06	5.00	-1.45	95.00	99.87	179.38	180.00	0.00	0.13	0.13	0.62	0.00
-74.00	0.00	360.00	0.00	52.6	5.00	179.35	-0.01	180.00	5.00	-0.74	5.00	-0.57	5.00	-0.57	5.00	-0.74	94.50	99.88	178.75	180.00	0.00	0.12	0.12	1.25	0.00
-73.00	0.00	360.00	0.00	52.7	5.00	180.57	0.00	180.00	5.00	0.55	5.00	0.60	5.00	0.60	5.00	0.55	94.00	99.89	179.35	180.00	0.00	0.11	0.11	0.65	0.00
-72.00	0.00	360.00	0.00	52.8	5.00	181.20	0.00	180.00	5.00	1.21	5.00	1.20	5.00	1.20	5.00	1.21	93.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-71.00	0.00	360.00	0.00	52.9	5.00	180.61	0.00	180.00	5.00	0.62	5.00	0.60	5.00	0.60	5.00	0.62	93.00	99.87	181.20	180.00	0.00	0.13	0.13	-1.20	0.00
-70.00	0.00	360.00	0.00	53	5.00	179.38	0.00	180.00	5.00	-0.61	5.00	-0.63	5.00	-0.63	5.00	-0.61	92.50	99.87	180.61	180.00	0.00	0.13	0.13	-0.61	0.00
-69.00	0.00	360.00	0.00	53.1	5.00	178.75	0.00	180.00	5.00	-1.25	5.00	-1.26	5.00	-1.26	5.00	-1.25	92.00	99.87	179.38	180.00	0.00	0.13	0.13	0.62	0.00
-68.00	0.00	360.00	0.00	53.2	5.00	179.34	0.00	180.00	5.00	-0.66	5.00	-0.66	5.00	-0.66	5.00	-0.66	91.50	99.89	178.75	180.00	0.00	0.11	0.11	1.25	0.00
-67.00	0.00	360.00	0.00	53.3	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	91.00	99.89	179.34	180.00	0.00	0.11	0.11	0.66	0.00
-66.00	0.00	360.00	0.00	53.4	5.00	181.21	0.00	180.00	5.00	1.21	5.00	1.21	5.00	1.21	5.00	1.21	90.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-65.00	0.00	360.00	0.00	53.5	5.00	180.62	0.00	180.00	5.00	0.62	5.00	0.62	5.00	0.62	5.00	0.62	90.00	99.88	181.21	180.00	0.00	0.12	0.12	-1.21	0.00
-64.00	0.00	360.00	0.00	53.6	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	89.50	99.87	180.62	180.00	0.00	0.13	0.13	-0.62	0.00
-63.00	0.00	360.00	0.00	53.7	5.00	178.74	0.00	180.00	5.00	-1.26	5.00	-1.26	5.00	-1.26	5.00	-1.26	89.00	99.88	179.38	180.00	0.00	0.12	0.12	0.62	0.00
-62.00	0.00	360.00	0.00	53.8	5.00	179.33	0.00	180.00	5.00	-0.67	5.00	-0.67	5.00	-0.67	5.00	-0.67	88.50	99.89	178.74	180.00	0.00	0.11	0.11	1.26	0.00
-61.00	0.00	360.00	0.00	53.9	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	88.00	99.89	179.33	180.00	0.00	0.11	0.11	0.67	0.00
-60.00	0.00	360.00	0.00	54	5.00	181.22	0.00	180.00	5.00	1.22	5.00	1.22	5.00	1.22	5.00	1.22	87.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-59.00	0.00	360.00	0.00	54.1	5.00	180.62	0.00	180.00	5.00	0.62	5.00	0.62	5.00	0.62	5.00	0.62	87.00	99.88	181.22	180.00	0.00	0.12	0.12	-1.22	0.00
-58.00	0.00	360.00	0.00	54.2	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	86.50	99.87	180.62	180.00	0.00	0.13	0.13	-0.62	0.00
-57.00	0.00	360.00	0.00	54.3	5.00	178.73	0.00	180.00	5.00	-1.27	5.00	-1.27	5.00	-1.27	5.00	-1.27	86.00	99.88	179.38	180.00	0.00	0.12	0.12	0.62	0.00
-56.00	0.00	360.00	0.00	54.4	5.00	179.33	0.00	180.00	5.00	-0.67	5.00	-0.67	5.00	-0.67	5.00	-0.67	85.50	99.89	178.73	180.00	0.00	0.11	0.11	1.27	0.00
-55.00	0.00	360.00	0.00	54.5	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	85.00	99.89	179.33	180.00	0.00	0.11	0.11	0.67	0.00
-54.00	0.00	360.00	0.00	54.6	5.00	181.22	0.00	180.00	5.00	1.22	5.00	1.22	5.00	1.22	5.00	1.22	84.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-53.00	0.00	360.00	0.00	54.7	5.00	180.63	0.00	180.00	5.00	0.63	5.00	0.63	5.00	0.63	5.00	0.63	84.00	99.88	181.22	180.00	0.00	0.12	0.12	-1.22	0.00
-52.00	0.00	360.00	0.00	54.8	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	83.50	99.87	180.63	180.00	0.00	0.13	0.13	-0.63	0.00
-51.00	0.00	360.00	0.00	54.9	5.00	178.73	0.00	180.00	5.00	-1.27	5.00	-1.27	5.00	-1.27	5.00	-1.27	83.00	99.88	179.38	180.00	0.00	0.12	0.12	0.62	0.00
-50.00	0.00	360.00	0.00	55	5.00	179.32	0.00	180.00	5.00	-0.68	5.00	-0.68	5.00	-0.68	5.00	-0.68	82.50	99.89	178.73	180.00	0.00	0.11	0.11	1.27	0.00
-49.00	0.00	360.00	0.00	55.1	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	82.00	99.90	179.32	180.00	0.00	0.10	0.10	0.68	0.00
-48.00	0.00	360.00	0.00	55.2	5.00	181.23	0.00	180.00	5.00	1.23	5.00	1.23	5.00	1.23	5.00	1.23	81.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-47.00	0.00	360.00	0.00	55.3	5.00	180.64	0.00	180.00	5.00	0.64	5.00	0.64	5.00	0.64	5.00	0.64	81.00	99.88	181.23	180.00	0.00	0.12	0.12	-1.23	0.00
-46.00	0.00	360.00	0.00	55.4	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	80.50	99.87	180.64	180.00	0.00	0.13	0.13	-0.64	0.00
-45.00	0.00	360.00	0.00	55.5	5.00	178.72	0.00	180.00	5.00	-1.28	5.00	-1.28	5.00	-1.28	5.00	-1.28	80.00	99.88	179.38	180.00	0.00	0.12	0.12	0.62	0.00
-44.00	0.00	360.00	0.00	55.6	5.00	179.32	0.00	180.00	5.00	-0.68	5.00	-0.68	5.00	-0.68	5.00	-0.68	79.50	99.89	178.72	180.00	0.00	0.11	0.11	1.28	0.00
-43.00	0.00	360.00	0.00	55.7	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	79.00	99.90	179.32	180.00	0.00	0.10	0.10	0.68	0.00
-42.00	0.00	360.00	0.00	55.8	5.00	181.24	0.00	180.00	5.00	1.24	5.00	1.24	5.00	1.24	5.00	1.24	78.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-41.00	0.00	360.00	0.00	55.9	5.00	180.64	0.00	180.00	5.00	0.64	5.00	0.64	5.00	0.64	5.00	0.64	78.00	99.88	181.24	180.00	0.00	0.12	0.12	-1.24	0.00
-40.00	0.00	360.00	0.00	56	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	77.50	99.88	180.64	180.00	0.00	0.12	0.12	-0.64	0.00
-39.00	0.00	360.00	0.00	56.1	5.00	178.72	0.00	180.00	5.00	-1.28	5.00	-1.28	5.00	-1.28	5.00	-1.28	77.00	99.88	179.38	180.00	0.00	0.12	0.12	0.62	0.00
-38.00	0.00	360.00	0.00	56.2	5.00	179.31	0.00	180.00	5.00	-0.69	5.00	-0.69	5.00	-0.69	5.00	-0.69	76.50	99.89	178.72	180.00	0.00	0.11	0.11	1.28	0.00
-37.00	0.00	360.00	0.00	56.3	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	76.00	99.90	179.31	180.00	0.00	0.10	0.10	0.69	0.00
-36.00	0.00	360.00	0.00	56.4	5.00	181.24	0.00	180.00	5.00	1.24	5.00	1.24	5.00	1.24	5.00	1.24	75.50	99.89	180.57	180.00	0.00	0.11	0.11	-0.57	0.00
-35.00	0.00	360.00	0.00	56.5	5.00	180.65	0.00	180.00	5.00	0.65	5.00	0.65	5.00	0.65	5.00	0.65	75.00	99.88	181.24	180.00	0.00	0.12	0.12	-1.24	0.00
-34.00	0.00	360.00	0.00	56.6	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	74.50	99.88	180.65	180.00	0.00	0.12	0.12	-0.65	0.00
-33.00	0.00	360.00	0.00	56.7	5.00	178.71	0.00	180.00	5.00	-1.29	5.00	-1.29	5.00	-1.29	5.00	-1.29	74.00	99.88	179.38	180.00	0.00	0.12	0.12	0.62	0.00
-32.00	0.00	360.00	0.00	56.8	5.00	179.30	0.00	180.00	5.00	-0.70	5.00	-0.70	5.00	-0.70	5.00	-0.70	73.51	99.89	178.71	180.00	-0.01	0.11	0.11	1.29	0.00
-31.00	0.00	360.00	0.00	56.9	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	73.01	99.90	179.30	180.00	-0.01	0.10	0.10	0.70	0.00
-30.00	0.00	360.00	0.00	57	5.00	181.25	0.00	180.00	5.00	1.25	5.00	1.25	5.00	1.25	5.00	1.25	72.51	99.89	180.57	180.00	-0.01	0.11	0.11	-0.57	0.00
-29.00	0.00	360.00	0.00	57.1	5.00	180.66	0.00	180.00	5.00	0.66	5.00	0.66	5.00	0.66	5.00	0.66	72.01	99.88	181.25	180.00	-0.01	0.12	0.12	-1.25	0.00
-28.00	0.00	360.00	0.00	57.2	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	71.51	99.88	180.66	180.00	-0.01	0.12	0.12	-0.66	0.00
-27.00	0.00	360.00	0.00	57.3	5.00	178.70	0.00	180.00	5.00	-1.30	5.00	-1.30	5.00	-1.30	5.00	-1.30	71.01	99.88	179.38	180.00	-0.01	0.12	0.12	0.62	0.00
-26.00	0.00	360.00	0.00	57.4	5.00	179.30	0.00	180.00	5.00	-0.70	5.00	-0.70	5.00	-0.70	5.00	-0.70	70.51	99.89	178.70	180.00	-0.01	0.11	0.11	1.30	0.00
-25.00	0.00	360.00	0.00	57.5	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	70.01	99.90	179.30	180.00	-0.01	0.10	0.10	0.70	0.00
-24.00	0.00	360.00	0.00	57.6	5.00	181.26	0.00	180.00	5.00	1.26	5.00	1.26	5.00	1.26	5.00	1.26	69.51	99.90	180.57	180.00	-0.01	0.10	0.11	-0.57	0.00
-23.00	0.00	360.00	0.00	57.7	5.00	180.66	0.00	180.00	5.00	0.66	5.00	0.66	5.00	0.66	5.00	0.66	69.01	99.88	181.26	180.00	-0.01	0.12	0.12	-1.26	0.00
-22.00	0.00	360.00	0.00	57.8	5.00	179.38	0.00	180.00	5.00	-0.62	5.00	-0.62	5.00	-0.62	5.00	-0.62	68.51	99.88	180.66	180.00	-0.01	0.12	0.12	-0.66	0.00
-21.00	0.00	360.00	0.00	57.9	5.00	178.70	0.00	180.00	5.00	-1.30	5.00	-1.30	5.00	-1.30	5.00	-1.30	68.01	99.88	179.38	180.00	-0.01	0.12	0.12	0.62	0.00
-20.00	0.00	360.00	0.00	58	5.00	179.29	0.00	180.00	5.00	-0.71	5.00	-0.71	5.00	-0.71	5.00	-0.71	67.51	99.90	178.70	180.00	-0.01	0.10	0.11	1.30	0.00
-19.00	0.00	360.00	0.00	58.1	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	67.01	99.90	179.29	180.00	-0.01	0.10	0.10	0.71	0.00
-18.00	0.00	360.00	0.00	58.2	5.00	181.26	0.00	180.00	5.00	1.26	5.00	1.26	5.00	1.26	5.00	1.26	66.51	99.90	180.57	180.00	-0.01	0.10	0.10	-0.57	0.00
-17.00	0.00	360.00	0.00	58.3	5.00	180.67	0.00	180.00	5.00	0.67	5.00	0.67	5.00	0.67	5.00	0.67	66.01	99.89	181.26	180.00	-0.01	0.11	0.11	-1.26	0.00
-16.00	0.00	360.00	0.00	58.4	5.00	179.39	0.00	180.00	5.00	-0.61	5.00	-0.61	5.00	-0.61	5.00	-0.61	65.51	99.88	180.67	180.00	-0.01	0.12	0.12	-0.67	0.00
-15.00	0.00	360.00	0.00	58.5	5.00	178.69	0.00	180.00	5.00	-1.31	5.00	-1.31	5.00	-1.31	5.00	-1.31	65.01	99.88	179.39	180.00	-0.01	0.12	0.12	0.61	0.00
-14.00	0.00	360.00	0.00	58.6	5.00	179.28	0.00	180.00	5.00	-0.72	5.00	-0.72	5.00	-0.72	5.00	-0.72	64.51	99.90	178.69	180.00	-0.01	0.10	0.10	1.31	0.00
-13.00	0.00	360.00	0.00	58.7	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	64.01	99.90	179.28	180.00	-0.01	0.10	0.10	0.72	0.00
-12.00	0.00	360.00	0.00	58.8	5.00	181.27	0.00	180.00	5.00	1.27	5.00	1.27	5.00	1.27	5.00	1.27	63.51	99.90	180.57	180.00	-0.01	0.10	0.10	-0.57	0.00
-11.00	0.00	360.00	0.00	58.9	5.00	180.68	0.00	180.00	5.00	0.68	5.00	0.68	5.00	0.68	5.00	0.68	63.01	99.89	181.27	180.00	-0.01	0.11	0.11	-1.27	0.00
-10.00	0.00	360.00	0.00	59	5.00	179.39	0.00	180.00	5.00	-0.61	5.00	-0.61	5.00	-0.61	5.00	-0.61	62.51	99.88	180.68	180.00	-0.01	0.12	0.12	-0.68	0.00
-9.00	0.00	360.00	0.00	59.1	5.00	178.68	0.00	180.00	5.00	-1.32	5.00	-1.32	5.00	-1.32	5.00	-1.32	62.01	99.89	179.39	180.00	-0.01	0.11	0.11	0.61	0.00
-8.00	0.00	360.00	0.00	59.2	5.00	179.28	0.00	180.00	5.00	-0.72	5.00	-0.72	5.00	-0.72	5.00	-0.72	61.51	99.90	178.68	180.00	-0.01	0.10	0.10	1.32	0.00

-7.00	0.00	360.00	0.00
-6.00	0.00	360.00	0.00
-5.00	0.00	360.00	0.00
-4.00	0.00	360.00	0.00
-3.00	0.00	360.00	0.00
-2.00	0.00	360.00	0.00
-1.00	0.00	360.00	0.00
0.00	0.00	360.00	0.00

59.3	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	61.01	99.90	179.28	180.00	-0.01	0.10	0.10	0.72	0.00
59.4	5.00	181.28	0.00	180.00	5.00	1.28	5.00	1.28	5.00	1.28	5.00	1.28	60.51	99.90	180.57	180.00	-0.01	0.10	0.10	-0.57	0.00
59.5	5.00	180.69	0.00	180.00	5.00	0.69	5.00	0.69	5.00	0.69	5.00	0.69	60.01	99.89	181.28	180.00	-0.01	0.11	0.11	-1.28	0.00
59.6	5.00	179.39	0.00	180.00	5.00	-0.61	5.00	-0.61	5.00	-0.61	5.00	-0.61	59.51	99.88	180.69	180.00	-0.01	0.12	0.12	-0.69	0.00
59.7	5.00	178.68	0.00	180.00	5.00	-1.32	5.00	-1.32	5.00	-1.32	5.00	-1.32	59.01	99.89	179.39	180.00	-0.01	0.11	0.11	0.61	0.00
59.8	5.00	179.27	0.00	180.00	5.00	-0.73	5.00	-0.73	5.00	-0.73	5.00	-0.73	58.51	99.90	178.68	180.00	-0.01	0.10	0.10	1.32	0.00
59.9	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	58.01	99.90	179.27	180.00	-0.01	0.10	0.10	0.73	0.00
60	5.00	181.29	0.00	180.00	5.00	1.29	5.00	1.29	5.00	1.29	5.00	1.29	57.51	99.90	180.57	180.00	-0.01	0.10	0.10	-0.57	0.00
60.1	5.00	180.69	0.00	180.00	5.00	0.69	5.00	0.69	5.00	0.69	5.00	0.69	57.01	99.89	181.29	180.00	-0.01	0.11	0.11	-1.29	0.00
60.2	5.00	179.39	0.00	180.00	5.00	-0.61	5.00	-0.61	5.00	-0.61	5.00	-0.61	56.51	99.88	180.69	180.00	-0.01	0.12	0.12	-0.69	0.00
60.3	5.00	178.67	0.00	180.00	5.00	-1.33	5.00	-1.33	5.00	-1.33	5.00	-1.33	56.01	99.89	179.39	180.00	-0.01	0.11	0.11	0.61	0.00
60.4	5.00	179.26	0.00	180.00	5.00	-0.74	5.00	-0.74	5.00	-0.74	5.00	-0.74	55.51	99.90	178.67	180.00	-0.01	0.10	0.10	1.33	0.00
60.5	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	55.01	99.91	179.26	180.00	-0.01	0.09	0.09	0.74	0.00
60.6	5.00	181.29	0.00	180.00	5.00	1.29	5.00	1.29	5.00	1.29	5.00	1.29	54.51	99.90	180.57	180.00	-0.01	0.10	0.10	-0.57	0.00
60.7	5.00	180.70	0.00	180.00	5.00	0.70	5.00	0.70	5.00	0.70	5.00	0.70	54.01	99.89	181.29	180.00	-0.01	0.11	0.11	-1.29	0.00
60.8	5.00	179.39	0.00	180.00	5.00	-0.61	5.00	-0.61	5.00	-0.61	5.00	-0.61	53.51	99.88	180.70	180.00	-0.01	0.12	0.12	-0.70	0.00
60.9	5.00	178.66	0.00	180.00	5.00	-1.34	5.00	-1.34	5.00	-1.34	5.00	-1.34	53.01	99.89	179.39	180.00	-0.01	0.11	0.11	0.61	0.00
61	5.00	179.26	0.00	180.00	5.00	-0.74	5.00	-0.74	5.00	-0.74	5.00	-0.74	52.51	99.90	178.66	180.00	-0.01	0.10	0.10	1.34	0.00
61.1	5.00	180.57	0.00	180.00	5.00	0.57	5.00	0.57	5.00	0.57	5.00	0.57	52.01	99.91	179.26	180.00	-0.01	0.09	0.09	0.74	0.00
61.2	5.00	181.30	0.00	180.00	5.00	1.30	5.00	1.30	5.00	1.30	5.00	1.30	51.51	99.90	180.57	180.00	-0.01	0.10	0.10	-0.57	0.00
61.3	5.00	180.71	0.00	180.00	5.00	0.71	5.00	0.71	5.00	0.71	5.00	0.71	51.01	99.89	181.30	180.00	-0.01	0.11	0.11	-1.30	0.00
61.4	5.00	179.39	0.00	180.00	5.00	-0.61	5.00	-0.61	5.00	-0.61	5.00	-0.61	50.51	99.88	180.71	180.00	-0.01	0.12	0.12	-0.71	0.00
61.5	5.00	178.66	0.00	180.00	5.00	-1.34	5.00	-1.34	5.00	-1.34	5.00	-1.34	50.01	99.89	179.39	180.00	-0.01	0.11	0.11	0.61	0.00
61.6	5.00	179.25	0.00	180.00	5.00	-0.75	5.00	-0.75	5.00	-0.75	5.00	-0.75	49.51	99.90	178.66	180.00	-0.01	0.10	0.10	1.34	0.90
61.7	5.00	180.57	2.70	180.00	5.01	31.86	4.96	-30.88	6.28	-23.92	6.32	24.74	49.01	99.91	179.25	180.00	-0.01	0.09	0.09	0.75	1.80
61.8	5.00	181.31	5.40	180.27	6.35	56.36	6.20	-55.43	8.25	-38.20	8.36	39.18	48.51	99.90	180.57	180.27	-0.01	0.10	0.10	-0.57	2.43
61.9	5.00	180.72	7.29	180.81	7.62	66.82	7.63	-66.87	9.91	-45.06	9.90	44.99	48.01	99.89	181.31	180.81	-0.01	0.11	0.11	-1.31	2.79
62	5.00	179.39	8.37	181.54	8.31	71.06	8.67	-71.87	11.00	-48.46	10.72	47.13	47.51	99.89	180.72	181.54	-0.01	0.11	0.12	-0.72	2.96
62.1	5.00	178.65	8.88	182.38	8.60	72.76	9.23	-73.96	11.57	-50.04	11.08	47.88	47.01	99.89	179.39	182.38	-0.01	0.11	0.11	0.61	3.02
62.2	5.00	179.24	9.07	183.26	8.74	73.40	9.41	-74.62	11.76	-50.50	11.23	48.22	46.51	99.90	178.65	183.26	-0.01	0.10	0.10	1.35	3.04
62.3	5.00	180.57	9.11	184.17	8.80	73.58	9.41	-74.66	11.76	-50.44	11.29	48.41	46.01	99.91	179.24	184.17	-0.01	0.09	0.09	0.76	3.03
62.4	5.00	181.31	9.09	185.08	8.77	73.48	9.40	-74.62	11.76	-50.45	11.26	48.32	45.51	99.90	180.57	185.08	-0.01	0.10	0.10	-0.57	3.02
62.5	5.00	180.72	9.05	185.99	8.61	73.19	9.50	-74.79	11.82	-50.83	11.12	47.84	45.01	99.89	181.31	185.99	-0.01	0.11	0.11	-1.31	3.01
62.6	5.00	179.39	9.03	186.90	8.40	72.85	9.66	-75.14	11.93	-51.45	10.94	47.18	44.51	99.89	180.72	186.90	-0.01	0.11	0.11	-0.72	3.00
62.7	5.00	178.65	9.01	187.80	8.25	72.64	9.78	-75.42	12.02	-51.92	10.81	46.72	44.01	99.89	179.39	187.80	-0.01	0.11	0.11	0.61	3.00
62.8	5.00	179.24	9.00	188.70	8.21	72.58	9.79	-75.46	12.03	-52.00	10.78	46.61	43.51	99.90	178.65	188.70	-0.01	0.10	0.10	1.35	3.00
62.9	5.00	180.57	9.00	189.60	8.25	72.60	9.75	-75.36	12.00	-51.86	10.81	46.72	43.01	99.91	179.24	189.60	-0.01	0.09	0.09	0.76	3.00
63	5.00	181.32	9.00	190.50	8.23	72.59	9.77	-75.39	12.01	-51.91	10.80	46.68	42.51	99.90	180.57	190.50	-0.01	0.10	0.10	-0.57	3.00
63.1	5.00	180.73	9.00	191.40	8.10	72.47	9.88	-75.70	12.10	-52.36	10.69	46.29	42.01	99.89	181.32	191.40	-0.01	0.11	0.11	-1.32	3.00
63.2	5.00	179.39	9.00	192.30	7.91	72.33	10.06	-76.19	12.22	-53.06	10.52	45.73	41.51	99.89	180.73	192.30	-0.01	0.11	0.11	-0.73	3.00
63.3	5.00	178.64	9.00	193.20	7.77	72.25	10.19	-76.57	12.32	-53.58	10.40	45.33	41.01	99.89	179.39	193.20	-0.01	0.11	0.11	0.61	3.00
63.4	5.00	179.23	9.00	194.10	7.74	72.24	10.21	-76.64	12.33	-53.68	10.38	45.25	40.51	99.90	178.64	194.10	-0.01	0.10	0.10	1.36	3.00
63.5	5.00	180.57	9.00	195.00	7.78	72.26	10.18	-76.54	12.31	-53.54	10.41	45.36	40.01	99.91	179.23	195.00	-0.01	0.09	0.09	0.77	3.00
63.6	5.00	181.33	9.00	195.90	7.77	72.26	10.19	-76.57	12.32	-53.58	10.40	45.33	39.51	99.91	180.57	195.90	-0.01	0.09	0.09	-0.57	3.00
63.7	5.00	180.74	9.00	196.80	7.64	72.21	10.30	-76.92	12.40	-54.06	10.29	44.97	39.01	99.89	181.33	196.80	-0.01	0.11	0.11	-1.33	3.00
63.8	5.00	179.39	9.00	197.70	7.44	72.20	10.47	-77.46	12.52	-54.77	10.11	44.45	38.51	99.89	180.74	197.70	-0.01	0.11	0.11	-0.74	3.00
63.9	5.00	178.63	9.00	198.60	7.29	72.22	10.60	-77.87	12.60	-55.31	9.98	44.08	38.01	99.89	179.39	198.60	-0.01	0.11	0.11	0.61	3.00
64	5.00	179.22	9.00	199.50	7.27	72.23	10.62	-77.95	12.62	-55.41	9.96	44.02	37.51	99.91	178.63	199.50	-0.01	0.09	0.10	1.37	3.00
64.1	5.00	180.57	9.00	200.40	7.31	72.22	10.59	-77.84	12.59	-55.26	10.00	44.11	37.01	99.91	179.22	200.40	-0.01	0.09	0.09	0.78	3.00
64.2	5.00	181.33	9.00	201.30	7.29	72.22	10.60	-77.87	12.60	-55.31	9.98	44.08	36.51	99.91	180.57	201.30	-0.01	0.09	0.09	-0.57	3.00
64.3	5.00	180.75	9.00	202.20	7.17	72.28	10.71	-78.25	12.68	-55.79	9.87	43.76	36.01	99.90	181.33	202.20	-0.01	0.10	0.11	-1.33	3.00
64.4	5.00	179.39	9.00	203.10	6.97	72.41	10.87	-78.83	12.78	-56.53	9.69	43.30	35.51	99.89	180.75	203.10	-0.01	0.11	0.11	-0.75	3.00
64.5	5.00	178.63	9.00	204.00	6.82	72.56	10.99	-79.27	12.86	-57.08	9.55	42.97	35.01	99.89	179.39	204.00	-0.01	0.11	0.11	0.61	3.00
64.6	5.00	179.21	9.00	204.90	6.80	72.59	11.01	-79.36	12.88	-57.18	9.53	42.91	34.51	99.91	178.63	204.90	-0.01	0.09	0.09	1.37	3.00
64.7	5.00	180.57	9.00	205.80	6.84	72.55	10.98	-79.23	12.86	-57.03	9.56	43.00	34.01	99.91	179.21	205.80	-0.01	0.09	0.09	0.79	3.00
64.8	5.00	181.34	9.00	206.70	6.83	72.56	10.99	-79.27	12.86	-57.07	9.55	42.97	33.51	99.91	180.57	206.70	-0.01	0.09	0.09	-0.57	3.00
64.9	5.00	180.75	9.00	207.60	6.70	72.73	11.09	-79.67	12.93	-57.57	9.43	42.69	33.01	99.90	181.34	207.60	-0.01	0.10	0.10	-1.34	3.00
65	5.00	179.39	9.00	208.50	6.50	73.05	11.25	-80.29	13.03	-58.32	9.25	42.28	32.51	99.89	180.75	208.50	-0.01	0.11	0.11	-0.75	3.00
65.1	5.00	178.62	9.00	209.40	6.36	73.35	11.36	-80.76	13.10	-58.88	9.11	42.00	32.01	99.89	179.39	209.40	-0.01	0.11	0.11	0.61	3.00
65.2	5.00	179.21	9.00	210.30	6.34	73.41	11.38	-80.85	13.11	-58.99	9.08	41.95	31.51	99.91	178.62	210.30	-0.01	0.09	0.09	1.38	3.00
65.3	5.00	180.57	9.00	211.20	6.37	73.32	11.35	-80.72	13.09	-58.83	9.12	42.03	31.01	99.91	179.21	211.20	-0.01	0.09	0.09	0.79	3.00
65.4	5.00	181.35	9.00	212.10	6.36	73.34	11.36	-80.76	13.10	-58.87	9.11	42.01	30.51	99.91	180.57	212.10	-0.01	0.09	0.09	-0.57	3.00
65.5	5.00	180.76	9.00	213.00	6.24	73.65	11.46	-81.18	13.16	-59.38	8.99	41.77	30.01	99.90	181.35	213.00	-0.01	0.10	0.10	-1.35	3.00
65.6	5.00	179.39	9.00	213.90	6.05	74.19	11.60	-81.84	13.24	-60.14	8.80	41.44	29.51	99.89	180.76	213.90	-0.01	0.11	0.11	-0.76	3.00
65.7	5.00	178.61	9.00	214.80	5.91	74.67	11.71	-82.33	13.31	-60.72	8.65	41.22	29.01	99.90	179.39	214.80	-0.01	0.10	0.10	0.61	3.00
65.8	5.00	179.20	9.00	215.70	5.89	74.77	11.73	-82.42	13.32	-60.83	8.63	41.18	28.51	99.91	178.61	215.70	-0.01	0.09	0.09	1.39	3.00
65.9	5.00	180.57	9.00	216.60	5.92	74.62	11.70	-82.28	13.30	-60.66	8.67	41.24	28.01	99.91	179.20	216.60	-0.01	0.09	0.09	0.80	3.00
66	5.00	181.36	9.00	217.50	5.92	74.66	11.71	-82.32	13.30	-60.70	8.66	41.23	27.51	99.91	180.57	217.50	-0.01	0.09	0.09	-0.57	3.00
66.1	5.00	180.77	9.00	218.40	5.80	75.13	11.80	-82.76	13.36	-61.21	8.53	41.05	27.01	99.90	181.36	218.40	-0.01	0.10	0.10	-1.36	3.00
66.2	5.00	179.40	9.00	219.30	5.61	75.95	11.94	-83.44	13.43	-61.99	8.33	40.81	26.51	99.89	180.77	219.30	-0.01	0.11	0.11	-0.77	3.00
66.3	5.00	178.61	9.00	220.20	5.48	76.64	12.04	-83.96	13.49	-62.58	8.19	40.65	26.01	99.90	179.40	220.20	-0.01	0.10	0.10	0.60	3.00
66.4	5.00	179.19	9.00	221.10	5.46	76.78	12.06	-84.06	13.50	-62.69	8.16	40.63	25.51	99.91	178.61	221.10	-0.01	0.09	0.09	1.39	3.00
66.5	5.00	180.57	9.00	222.00	5.50	76.57	12.03	-83.91	13.48	-62.52	8.20	40.67	25.01	99.92	179.19	222.00	-0.01	0.08	0.08	0.81	3.00
66.6	5.00	181.36	9.00	222.90	5.49	76.62	12.04	-83.94	13.49	-62.56	8.19	40.66	24.51	99.91	180.57	222.90	-0.01	0.09	0.09	-0.57	3.00
66.7	5.00	180.78	9.00	223.80	5.37	77.28	12.12	-84.40	13.53	-63.07	8.06	40.55	24.01	99.90	181.36	223.80	-0.01	0.10	0.10	-1.36	3.00
66.8	5.00	179.40	9.00	224.70	5.21	78.43	12.25	-85.11	13.60	-63.87	7.87	40.41	23.51	99.89	180.78	224.70	-0.01	0.11	0.11	-0.78	3.00
66.9	5.00	178.60	9.00	225.60	5.08	79.37	12.35	-85.65	13.64	-64.46	7.72	40.35	23.01	99.90	179.40	225.60	-0.01	0.10	0.10	0.60	3.00
67	5.00	179.19	9.00	226.50	5.06	79.56	12.36	-85.74	13.65	-64.57	7.69	40.34	22.51	99.91	178.60	226.50	-0.01	0.09	0.09	1.40	3.00
67.1	5.00	180.57	9.00	227.40	5.10	79.28	12.34	-85.59	13.64	-64.40	7.73	40.35	22.01	99.92	179.19	227.40	-0.01	0.08	0.08	0.81	3.00
67.2	5.00	181.37	9.00	228.30	5.09	79.33	12.34	-85.62	13.64	-64.44	7.72	40.35	21.51	99.91	180.57	228.30	-0.01	0.09	0.09	-0.57	3.00
67.3	5.00	180.79	9.00	229.20	4.99	80.23	12.42	-86.09	13.68	-64.95	7.59	40.32	21.01	99.90	181.37	229.20	-0.01	0.10	0.10	-1.37	3.00
67.4	5.00	179.40	9.00	230.10	4.83	81.74	12.54	-86.83	13.73	-65.76	7.40	40.31	20.51	99.89	180.79	230.10	-0.01	0.11	0.11	-0.79	3.00
67.5	5.00	178.59	9.00	231.00	4.73	82.98	12.63	-87.38	13.77	-66.36	7.25	40.35	20.01	99.90	179.40	231.00	-0.01	0.10	0.10	0.60	3.00
67.6	5.00	179.18	9.00	231.90	4.71	83.22	12.64	-87.48	13.78	-66.47	7.22	40.36	19.51	99.91	178.59	231.90	-0.01	0.09	0.09	1.41	3.00
67.7	5.00	180.57	9.00	232.80	4.74	82.85	12.62	-87.32	13.77	-66.30	7.26	40.34	19.01	99.92	179.18	232.80	-0.01	0.08	0.08	0.82	3.00
67.8	5.00	181.38	9.00	233.70	4.73	82.92	12.62	-87.35	13.77	-66.33	7.25	40.35	18.51	99.91	180.57	233.70	-0.01	0.09	0.09	-0.57	3.00
67.9	5.00	180.79	9.00	234.60	4.64	84.07	12.70	-87.83	13.80	-66.85	7.12	40.41	18.01	99.90	181.38	234.60	-0.01	0.10	0.10	-1.38	3.00
68	5.00	179.40	9.00	235.50	4.51	85.98	12.81	-88.59	13.84	-67.66	6.93	40.56	17.51	99.89	180.79	235.50	-0.01	0.11	0.11	-0.79	3.00
68.1	5.00	178.59	9.00	236.40	4.43	87.53	12.89	-89.15	13.87	-68.27	6.78	40.73	17.01	99.90	179.40	236.40	-0.01	0.10	0.10	0.60	3.00
68.2	5.00	179.17	9.00	237.30	4.41	87.82	12.90	-89.26	13.88	-68.38	6.75	40.76	16.51	99.91	178.59	237.30	-0.01	0.09	0.09	1.41	3.00
68.3	5.00	180.57	9.00	238.20	4.44	87.36	12.88	-89.09	13.87	-68.20	6.79	40.71	16.01	99.92	179.17	238.20	-0.01	0.08	0.08	0.83	3.00
68.4	5.00	181.38	9.00	239.10	4.43	87.44	12.88	-89.12	13.87	-68.23	6.79	40.72	15.51	99.91	180.57	239.10	-0.01	0.09	0.09	-0.57	3.00
68.5	5.00	180.80	9.00	240.00	4.36	88.85	12.95	-89.61	13.89	-68.76	6.66	40.90	15.01	99.90	181.38	240.00	-0.01	0.10	0.10	-1.38	3.00
68.6	5.00	179.40	9.00	240.90	4.26	-88.83	13.05	89.62	13.92	-69.58	6.46	41.24	14.51	99.89	180.80	240.90	-0.01	0.11	0.11	-0.80	3.00
68.7	5.00	178.58	9.00	241.80	4.20	-87.00	13.12	89.04	13.94	-70.19	6.32	41.56	14.01	99.90	179.40	241.80	-0.01	0.10	0.10	0.60	3.00
68.8	5.00	179.16	9.00	242.70	4.18	-86.65	13.13	88.93	13.95	-70.30	6.29	41.63	13.51	99.91	178.58	242.70	-0.01	0.09	0.09	1.42	3.00
68.9	5.00	180.57	9.00	243.60	4.20	-87.20	13.11	89.10	13.94	-70.12	6.33	41.53	13.01	99.92	179.16	243.60	-0.01	0.08	0.08	0.84	3.00
69	5.00	181.39	9.00	244.50	4.20	-87.12	13.11	89.08	13.94	-70.15	6.32	41.54	12.51	99.91	180.57	244.50	-0.01	0.09	0.09	-0.57	3.00
69.1	5.00	180.81	9.00	245.40	4.15	-85.48	13.17	88.58	13.96	-70.68	6.20	41.86	12.01	99.90	181.39	245.40	-0.01	0.10	0.10	-1.39	3.00
69.2	5.00	179.40	9.00	246.30	4.09	-82.82	13.26	87.79	13.97	-71.50	6.01	42.44	11.51	99.90	180.81	246.30	-0.01	0.10	0.11	-0.81	3.00
69.3	5.00	178.57	9.00	247.20	4.05	-80.76	13.33	87.20	13.99	-72.12	5.87	42.95	11.01	99.90	179.40	247.20	-0.01	0.10	0.10	0.60	3.00
69.4	5.00	179.16	9.00	248.10	4.04	-80.38	13.34	87.09	13.99	-72.23	5.84	43.05	10.51	99.91	178.57	248.10	-0.01	0.09	0.09	1.43	3.00
69.5	5.00	180.57	9.00	249.00	4.05	-80.99	13.32	87.27	13.98	-72.05	5.88	42.89	10.01	99.92	179.16	249.00	-0.01	0.08	0.08	0.84	3.00
69.6	5.00	181.40	9.00	249.90	4.05	-80.91	13.32	87.25	13.98	-72.07	5.88	42.91	9.51	99.92	180.57	249.90	-0.01	0.08	0.09	-0.57	3.00
69.7	5.00	180.82	9.00	250.80	4.03	-79.11	13.37	86.74	13.99	-72.60	5.76	43.39	9.01	99.90	181.40	250.80	-0.01	0.10	0.10	-1.40	3.00
69.8	5.00	179.40	9.00	251.70	4.01	-76.25	13.45	85.94	14.00	-73.43	5.57	44.26	8.51	99.90	180.82	251.70	-0.01	0.10	0.10	-0.82	3.00
69.9	5.00	178.57	9.00	252.60	4.00	-74.08	13.51	85.34	14.00	-74.05	5.44	44.99	8.01	99.90	179.40	252.60	-0.01	0.10	0.10	0.60	3.00
70	5.00	179.15	9.00	253.50	4.00	-73.68	13.52	85.23	14.00	-74.16	5.42	45.13	7.51	99.91	178.57	253.50	-0.01	0.09	0.09	1.43	3.00
70.1	5.00	180.57	9.00	254.40	4.00	-74.33	13.50	85.41	14.00	-73.98	5.46	44.90	7.01	99.92	179.15	254.40	-0.01	0.08	0.08	0.85	3.00
70.2	5.00	181.41	9.00	255.30	4.00	-74.26	13.50	85.39	14.00	-74.00	5.45	44.93	6.51	99.92	180.57	255.30	-0.01	0.08	0.08	-0.57	3.00
70.3	5.00	180.83	9.00	256.20	4.00	-72.41	13.55	84.88	14.00	-74.53	5.34	45.61	6.01	99.90	181.41	256.20	-0.01	0.10	0.10	-1.41	3.00
70.4	5.00	179.40	9.00	257.10	4.02	-69.53	13.61	84.07	13.99	-75.36	5.17	46.81	5.51	99.90	180.83	257.10	-0.01	0.10	0.10	-0.83	3.00
70.5	5.00	178.56	9.00	258.00	4.05	-67.40	13.66	83.46	13.99	-75.98	5.05	47.81	5.01	99.90	179.40	258.00	-0.01	0.10	0.10	0.60	3.00
70.6	5.00	179.14	9.00	258.90	4.06	-67.01	13.67	83.35	13.98	-76.09	5.02	48.00	4.51	99.91	178.56	258.90	-0.01	0.09	0.09	1.44	3.00
70.7	5.00	180.57	9.00	259.80	4.05	-67.65	13.65	83.53	13.99	-75.90	5.06	47.69	4.01	99.92	179.14	259.80	-0.01	0.08	0.08	0.86	3.00
70.8	5.00	181.41	9.00	260.70	4.05	-67.59	13.65	83.51	13.99	-75.92	5.06	47.72	3.51	99.92	180.57	260.70	-0.01	0.08	0.08	-0.57	3.00
70.9	5.00	180.83	9.00	261.60	4.08	-65.81	13.69	82.99	13.98	-76.45	4.95	48.63	3.01	99.90	181.41	261.60	-0.01	0.10	0.10	-1.41	3.00
71	5.00	179.40	9.00	262.50	4.14	-63.11	13.75	82.17	13.96	-77.28	4.80	50.21	2.51	99.90	180.83	262.50	-0.01	0.10	0.10	-0.83	3.00
71.1	5.00	178.55	9.00	263.40	4.19	-61.16	13.78	81.56	13.94	-77.90	4.69	51.51	2.01	99.90	179.40	263.40	-0.01	0.10	0.10	0.60	3.00

71.2	5.00	179.13	9.00	264.30	4.21	-60.81	13.79	81.45	13.94	-78.02	4.67	51.76	1.52	99.92	178.55	264.30	-0.02	0.08	0.09	1.45	3.00
71.3	5.00	180.57	9.00	265.20	4.19	-61.39	13.78	81.63	13.95	-77.83	4.71	51.34	1.02	99.92	179.13	265.20	-0.02	0.08	0.08	0.87	3.00
71.4	5.00	181.42	9.00	266.10	4.19	-61.34	13.78	81.62	13.94	-77.85	4.70	51.38	0.52	99.92	180.57	266.10	-0.02	0.08	0.08	-0.57	3.00
71.5	5.00	180.84	9.00	267.00	4.24	-59.75	13.81	81.10	13.93	-78.37	4.62	52.55	0.02	99.91	181.42	267.00	-0.02	0.09	0.10	-1.42	3.00
71.6	5.00	179.41	9.00	267.90	4.34	-57.36	13.85	80.27	13.90	-79.20	4.49	54.54	-0.48	99.90	180.84	267.90	-0.02	0.10	0.10	-0.84	3.00
71.7	5.00	178.55	9.00	268.80	4.42	-55.68	13.88	79.65	13.87	-79.83	4.40	56.15	-0.98	99.90	179.41	268.80	-0.02	0.10	0.10	0.59	3.00
71.8	5.00	179.12	9.00	269.70	4.44	-55.38	13.88	79.53	13.87	-79.94	4.38	56.46	-1.48	99.92	178.55	269.70	-0.02	0.08	0.09	1.45	3.00
71.9	5.00	180.56	9.00	270.60	4.41	-55.88	13.88	79.72	13.88	-79.75	4.41	55.95	-1.98	99.92	179.12	270.60	-0.02	0.08	0.08	0.88	3.00
72	5.00	181.43	9.00	271.50	4.42	-55.85	13.88	79.71	13.87	-79.76	4.41	55.98	-2.48	99.92	180.56	271.50	-0.02	0.08	0.08	-0.56	3.00
72.1	5.00	180.85	9.00	272.40	4.49	-54.50	13.90	79.19	13.85	-80.28	4.34	57.41	-2.98	99.91	181.43	272.40	-0.02	0.09	0.09	-1.43	3.00
72.2	5.00	179.41	9.00	273.30	4.62	-52.50	13.93	78.35	13.81	-81.11	4.24	59.80	-3.48	99.90	180.85	273.30	-0.02	0.10	0.10	-0.85	3.00
72.3	5.00	178.54	9.00	274.20	4.72	-51.12	13.95	77.72	13.77	-81.74	4.18	61.71	-3.98	99.90	179.41	274.20	-0.02	0.10	0.10	0.59	3.00
72.4	5.00	179.12	9.00	275.10	4.74	-50.88	13.95	77.61	13.77	-81.85	4.17	62.07	-4.48	99.92	178.54	275.10	-0.02	0.08	0.08	1.46	3.00
72.5	5.00	180.56	9.00	276.00	4.71	-51.29	13.95	77.80	13.78	-81.66	4.18	61.47	-4.98	99.92	179.12	276.00	-0.02	0.08	0.08	0.88	3.00
72.6	5.00	181.44	9.00	276.90	4.71	-51.27	13.95	77.79	13.78	-81.67	4.18	61.50	-5.48	99.92	180.56	276.90	-0.02	0.08	0.08	-0.56	3.00
72.7	5.00	180.86	9.00	277.80	4.80	-50.18	13.96	77.27	13.74	-82.19	4.14	63.15	-5.98	99.91	181.44	277.80	-0.02	0.09	0.09	-1.44	3.00
72.8	5.00	179.41	9.00	278.70	4.96	-48.59	13.98	76.43	13.69	-83.01	4.08	65.88	-6.48	99.90	180.86	278.70	-0.02	0.10	0.10	-0.86	3.00
72.9	5.00	178.53	9.00	279.60	5.08	-47.51	13.99	75.80	13.64	-83.64	4.04	68.01	-6.98	99.90	179.41	279.60	-0.02	0.10	0.10	0.59	3.00
73	5.00	179.11	9.00	280.50	5.10	-47.32	13.99	75.68	13.64	-83.75	4.04	68.41	-7.48	99.92	178.53	280.50	-0.02	0.08	0.08	1.47	3.00
73.1	5.00	180.56	9.00	281.40	5.07	-47.64	13.99	75.88	13.65	-83.55	4.04	67.73	-7.98	99.93	179.11	281.40	-0.02	0.07	0.08	0.89	3.00
73.2	5.00	181.44	9.00	282.30	5.07	-47.63	13.99	75.87	13.65	-83.56	4.04	67.76	-8.48	99.92	180.56	282.30	-0.02	0.08	0.08	-0.56	3.00
73.3	5.00	180.87	9.00	283.20	5.17	-46.79	13.99	75.34	13.61	-84.08	4.02	69.56	-8.98	99.91	181.44	283.20	-0.02	0.09	0.09	-1.44	3.00
73.4	5.00	179.41	9.00	284.10	5.34	-45.58	14.00	74.50	13.54	-84.90	4.00	72.49	-9.48	99.90	180.87	284.10	-0.02	0.10	0.10	-0.87	3.00
73.5	5.00	178.53	9.00	285.00	5.48	-44.77	14.00	73.87	13.49	-85.52	4.00	74.72	-9.98	99.91	179.41	285.00	-0.02	0.09	0.10	0.59	3.00
73.6	5.00	179.10	9.00	285.90	5.50	-44.63	14.00	73.75	13.48	-85.63	4.00	75.12	-10.48	99.92	178.53	285.90	-0.02	0.08	0.08	1.47	3.00
73.7	5.00	180.56	9.00	286.80	5.46	-44.87	14.00	73.95	13.50	-85.43	4.00	74.42	-10.98	99.93	179.10	286.80	-0.02	0.07	0.08	0.90	3.00
73.8	5.00	181.45	9.00	287.70	5.46	-44.87	14.00	73.95	13.50	-85.44	4.00	74.44	-11.48	99.92	180.56	287.70	-0.02	0.08	0.08	-0.56	3.00
73.9	5.00	180.88	9.00	288.60	5.58	-44.25	14.00	73.42	13.45	-85.95	4.01	76.28	-11.98	99.91	181.45	288.60	-0.02	0.09	0.09	-1.45	3.00
74	5.00	179.41	9.00	289.50	5.76	-43.37	13.99	72.57	13.37	-86.76	4.03	79.20	-12.48	99.90	180.88	289.50	-0.02	0.10	0.10	-0.88	3.00
74.1	5.00	178.52	9.00	290.40	5.91	-42.79	13.98	71.94	13.31	-87.38	4.06	81.37	-12.98	99.91	179.41	290.40	-0.02	0.09	0.10	0.59	3.00
74.2	5.00	179.09	9.00	291.30	5.94	-42.69	13.98	71.82	13.30	-87.49	4.07	81.76	-13.48	99.92	178.52	291.30	-0.02	0.08	0.08	1.48	3.00
74.3	5.00	180.56	9.00	292.20	5.89	-42.87	13.98	72.02	13.32	-87.29	4.06	81.08	-13.98	99.93	179.09	292.20	-0.02	0.07	0.08	0.91	3.00
74.4	5.00	181.46	9.00	293.10	5.89	-42.86	13.98	72.02	13.32	-87.30	4.06	81.09	-14.48	99.92	180.56	293.10	-0.02	0.08	0.08	-0.56	3.00
74.5	5.00	180.88	9.00	294.00	6.01	-42.44	13.97	71.49	13.26	-87.80	4.09	82.84	-14.98	99.91	181.46	294.00	-0.02	0.09	0.09	-1.46	3.00
74.6	5.00	179.41	9.00	294.90	6.21	-41.84	13.96	70.65	13.17	-88.60	4.15	85.57	-15.48	99.90	180.88	294.90	-0.02	0.10	0.10	-0.88	3.00
74.7	5.00	178.51	9.00	295.80	6.36	-41.46	13.94	70.01	13.10	-89.21	4.21	87.55	-15.98	99.91	179.41	295.80	-0.02	0.09	0.09	0.59	3.00
74.8	5.00	179.09	9.00	296.70	6.39	-41.40	13.93	69.89	13.09	-89.32	4.23	87.90	-16.48	99.92	178.51	296.70	-0.02	0.08	0.08	1.49	3.00
74.9	5.00	180.56	9.00	297.60	6.34	-41.51	13.94	70.10	13.11	-89.13	4.20	87.28	-16.98	99.93	179.09	297.60	-0.02	0.07	0.07	0.91	3.00
75	5.00	181.46	9.00	298.50	6.34	-41.51	13.94	70.10	13.11	-89.13	4.20	87.28	-17.48	99.92	180.56	298.50	-0.02	0.08	0.08	-0.56	3.00
75.1	5.00	180.89	9.00	299.40	6.46	-41.24	13.92	69.57	13.05	-89.62	4.26	88.84	-17.98	99.91	181.46	299.40	-0.02	0.09	0.09	-1.46	3.00
75.2	5.00	179.41	9.00	300.30	6.67	-40.89	13.89	68.73	12.94	89.58	4.36	-88.76	-18.48	99.90	180.89	300.30	-0.02	0.10	0.10	-0.89	3.00
75.3	5.00	178.51	9.00	301.20	6.82	-40.68	13.86	68.09	12.86	88.98	4.45	-87.06	-18.98	99.91	179.41	301.20	-0.02	0.09	0.09	0.59	3.00
75.4	5.00	179.08	9.00	302.10	6.85	-40.64	13.86	67.97	12.85	88.87	4.47	-86.76	-19.48	99.92	178.51	302.10	-0.02	0.08	0.08	1.49	3.00
75.5	5.00	180.56	9.00	303.00	6.80	-40.70	13.87	68.18	12.88	89.07	4.44	-87.29	-19.98	99.93	179.08	303.00	-0.02	0.07	0.07	0.92	3.00
75.6	5.00	181.47	9.00	303.90	6.80	-40.70	13.87	68.18	12.88	89.07	4.44	-87.30	-20.48	99.92	180.56	303.90	-0.02	0.08	0.08	-0.56	3.00
75.7	5.00	180.90	9.00	304.80	6.93	-40.56	13.84	67.66	12.81	88.59	4.51	-85.98	-20.98	99.91	181.47	304.80	-0.02	0.09	0.09	-1.47	3.00
75.8	5.00	179.41	9.00	305.70	7.13	-40.40	13.80	66.82	12.69	87.80	4.65	-83.99	-21.48	99.90	180.90	305.70	-0.02	0.10	0.10	-0.90	3.00
75.9	5.00	178.50	9.00	306.60	7.29	-40.33	13.76	66.18	12.60	87.21	4.76	-82.60	-21.98	99.91	179.41	306.60	-0.02	0.09	0.09	0.59	3.00
76	5.00	179.07	9.00	307.50	7.32	-40.33	13.75	66.06	12.59	87.11	4.78	-82.36	-22.48	99.92	178.50	307.50	-0.02	0.08	0.08	1.50	3.00
76.1	5.00	180.56	9.00	308.40	7.27	-40.34	13.77	66.27	12.62	87.30	4.74	-82.79	-22.98	99.93	179.07	308.40	-0.02	0.07	0.07	0.93	3.00
76.2	5.00	181.48	9.00	309.30	7.27	-40.34	13.77	66.28	12.62	87.30	4.74	-82.81	-23.48	99.92	180.56	309.30	-0.02	0.08	0.08	-0.56	3.00
76.3	5.00	180.91	9.00	310.20	7.39	-40.31	13.73	65.76	12.54	86.83	4.83	-81.75	-23.98	99.91	181.48	310.20	-0.02	0.09	0.09	-1.48	3.00
76.4	5.00	179.42	9.00	311.10	7.60	-40.32	13.68	64.92	12.42	86.06	4.99	-80.17	-24.48	99.90	180.91	311.10	-0.02	0.10	0.10	-0.91	3.00
76.5	5.00	178.49	9.00	312.00	7.76	-40.36	13.63	64.28	12.32	85.48	5.12	-79.08	-24.98	99.91	179.42	312.00	-0.02	0.09	0.09	0.58	3.00
76.6	5.00	179.06	9.00	312.90	7.79	-40.38	13.62	64.17	12.30	85.38	5.14	-78.89	-25.48	99.92	178.49	312.90	-0.02	0.08	0.08	1.51	3.00
76.7	5.00	180.56	9.00	313.80	7.74	-40.35	13.64	64.37	12.33	85.57	5.10	-79.23	-25.98	99.93	179.06	313.80	-0.02	0.07	0.07	0.94	3.00
76.8	5.00	181.49	9.00	314.70	7.74	-40.35	13.64	64.39	12.33	85.58	5.10	-79.25	-26.48	99.93	180.56	314.70	-0.02	0.07	0.08	-0.56	3.00
76.9	5.00	180.92	9.00	315.60	7.86	-40.41	13.60	63.87	12.25	85.12	5.20	-78.43	-26.98	99.91	181.49	315.60	-0.02	0.09	0.09	-1.49	3.00
77	5.00	179.42	9.00	316.50	8.07	-40.55	13.53	63.04	12.12	84.37	5.38	-77.23	-27.48	99.90	180.92	316.50	-0.02	0.10	0.10	-0.92	3.00
77.1	5.00	178.48	9.00	317.40	8.23	-40.70	13.47	62.40	12.01	83.80	5.52	-76.42	-27.98	99.91	179.42	317.40	-0.02	0.09	0.09	0.58	3.00
77.2	5.00	179.05	9.00	318.30	8.26	-40.73	13.46	62.29	11.99	83.70	5.55	-76.29	-28.48	99.92	178.48	318.30	-0.02	0.08	0.08	1.52	3.00
77.3	5.00	180.56	9.00	319.20	8.21	-40.67	13.48	62.50	12.03	83.89	5.50	-76.54	-28.98	99.93	179.05	319.20	-0.02	0.07	0.07	0.95	3.00
77.4	5.00	181.49	9.00	320.10	8.21	-40.67	13.48	62.51	12.03	83.90	5.50	-76.56	-29.48	99.93	180.56	320.10	-0.02	0.07	0.08	-0.56	3.00
77.5	5.00	180.93	9.00	321.00	8.33	-40.80	13.44	62.00	11.94	83.45	5.61	-75.96	-29.98	99.91	181.49	321.00	-0.02	0.09	0.09	-1.49	3.00
77.6	5.00	179.42	9.00	321.90	8.54	-41.06	13.35	61.17	11.79	82.73	5.80	-75.09	-30.48	99.90	180.93	321.90	-0.02	0.10	0.10	-0.93	3.00
77.7	5.00	178.48	9.00	322.80	8.70	-41.29	13.29	60.54	11.68	82.18	5.95	-74.52	-30.98	99.91	179.42	322.80	-0.02	0.09	0.09	0.58	3.00
77.8	5.00	179.04	9.00	323.70	8.72	-41.33	13.28	60.43	11.66	82.08	5.98	-74.42	-31.48	99.92	178.48	323.70	-0.02	0.08	0.08	1.52	3.00
77.9	5.00	180.56	9.00	324.60	8.67	-41.25	13.30	60.64	11.70	82.26	5.93	-74.60	-31.98	99.93	179.04	324.60	-0.02	0.07	0.07	0.96	3.00
78	5.00	181.50	9.00	325.50	8.67	-41.24	13.30	60.66	11.70	82.28	5.93	-74.61	-32.48	99.93	180.56	325.50	-0.02	0.07	0.08	-0.56	3.00
78.1	5.00	180.94	9.00	326.40	8.79	-41.44	13.24	60.16	11.61	81.84	6.05	-74.20	-32.98	99.91	181.50	326.40	-0.02	0.09	0.09	-1.50	3.00
78.2	5.00	179.42	9.00	327.30	9.00	-41.79	13.15	59.34	11.45	81.15	6.25	-73.62	-33.48	99.91	180.94	327.30	-0.02	0.09	0.10	-0.94	3.00
78.3	5.00	178.47	9.00	328.20	9.15	-42.09	13.08	58.71	11.33	80.62	6.41	-73.25	-33.98	99.91	179.42	328.20	-0.02	0.09	0.09	0.58	3.00
78.4	5.00	179.04	9.00	329.10	9.18	-42.14	13.06	58.60	11.30	80.53	6.43	-73.19	-34.48	99.92	178.47	329.10	-0.02	0.08	0.08	1.53	3.00
78.5	5.00	180.55	9.00	330.00	9.13	-42.04	13.09	58.81	11.34	80.70	6.38	-73.30	-34.98	99.93	179.04	330.00	-0.02	0.07	0.07	0.96	3.00
78.6	5.00	181.51	9.00	330.90	9.12	-42.03	13.09	58.83	11.35	80.72	6.38	-73.31	-35.48	99.93	180.55	330.90	-0.02	0.07	0.08	-0.55	3.00
78.7	5.00	180.94	9.00	331.80	9.24	-42.28	13.03	58.33	11.25	80.30	6.50	-73.06	-35.98	99.91	181.51	331.80	-0.02	0.09	0.09	-1.51	3.00
78.8	5.00	179.42	9.00	332.70	9.44	-42.71	12.92	57.53	11.08	79.64	6.71	-72.71	-36.48	99.91	180.94	332.70	-0.02	0.09	0.10	-0.94	3.00
78.9	5.00	178.46	9.00	333.60	9.59	-43.07	12.84	56.91	10.95	79.14	6.87	-72.51	-36.98	99.91	179.42	333.60	-0.02	0.09	0.09	0.58	3.00
79	5.00	179.03	9.00	334.50	9.62	-43.13	12.82	56.80	10.93	79.05	6.90	-72.48	-37.48	99.92	178.46	334.50	-0.02	0.08	0.08	1.54	3.00
79.1	5.00	180.55	9.00	335.40	9.57	-43.01	12.85	57.01	10.97	79.21	6.84	-72.54	-37.98	99.93	179.03	335.40	-0.02	0.07	0.07	0.97	3.00
79.2	5.00	181.52	9.00	336.30	9.57	-43.00	12.85	57.03	10.98	79.23	6.84	-72.55	-38.48	99.93	180.55	336.30	-0.02	0.07	0.08	-0.55	3.00
79.3	5.00	180.95	9.00	337.20	9.68	-43.29	12.79	56.54	10.87	78.84	6.97	-72.42	-38.98	99.91	181.52	337.20	-0.02	0.09	0.09	-1.52	3.00
79.4	5.00	179.42	9.00	338.10	9.88	-43.79	12.67	55.75	10.70	78.21	7.18	-72.27	-39.48	99.91	180.95	338.10	-0.02	0.09	0.10	-0.95	3.00
79.5	5.00	178.46	9.00	339.00	10.02	-44.20	12.58	55.14	10.56	77.74	7.34	-72.21	-39.98	99.91	179.42	339.00	-0.02	0.09	0.09	0.58	3.00
79.6	5.00	179.02	9.00	339.90	10.05	-44.27	12.56	55.04	10.53	77.66	7.37	-72.21	-40.48	99.93	178.46	339.90	-0.02	0.07	0.08	1.54	3.00
79.7	5.00	180.55	9.00	340.80	10.00	-44.13	12.59	55.24	10.58	77.82	7.31	-72.22	-40.98	99.93	179.02	340.80	-0.02	0.07	0.07	0.98	3.00
79.8	5.00	181.52	9.00	341.70	10.00	-44.12	12.59	55.26	10.59	77.83	7.31	-72.22	-41.48	99.93	180.55	341.70	-0.02	0.07	0.07	-0.55	3.00
79.9	5.00	180.96	9.00	342.60	10.11	-44.44	12.52	54.79	10.48	77.47	7.43	-72.20	-41.98	99.92	181.52	342.60	-0.02	0.08	0.09	-1.52	3.00
80	5.00	179.43	9.00	343.50	10.30	-45.00	12.39	54.01	10.29	76.89	7.65	-72.22	-42.48	99.91	180.96	343.50	-0.02	0.09	0.10	-0.96	3.00
80.1	5.00	178.45	9.00	344.40	10.44	-45.45	12.29	53.42	10.15	76.45	7.81	-72.28	-42.98	99.91	179.43	344.40	-0.02	0.09	0.09	0.57	3.00
80.2	5.00	179.01	9.00	345.30	10.46	-45.54	12.27	53.31	10.12	76.37	7.84	-72.29	-43.48	99.93	178.45	345.30	-0.02	0.07	0.08	1.55	3.00
80.3	5.00	180.55	9.00	346.20	10.42	-45.38	12.30	53.52	10.17	76.52	7.78	-72.26	-43.98	99.93	179.01	346.20	-0.02	0.07	0.07	0.99	3.00
80.4	5.00	181.53	9.00	347.10	10.41	-45.36	12.31	53.54	10.18	76.54	7.78	-72.26	-44.48	99.93	180.55	347.10	-0.02	0.07	0.07	-0.55	3.00
80.5	5.00	180.97	9.00	348.00	10.52	-45.72	12.23	53.08	10.07	76.21	7.90	-72.33	-44.98	99.92	181.53	348.00	-0.02	0.08	0.09	-1.53	3.00
80.6	5.00	179.43	9.00	348.90	10.70	-46.33	12.09	52.32	9.87	75.67	8.12	-72.48	-45.48	99.91	180.97	348.90	-0.02	0.09	0.10	-0.97	3.00
80.7	5.00	178.44	9.00	349.80	10.84	-46.82	11.98	51.74	9.72	75.28	8.28	-72.64	-45.98	99.91	179.43	349.80	-0.02	0.09	0.09	0.57	3.00
80.8	5.00	179.00	9.00	350.70	10.86	-46.91	11.96	51.64	9.70	75.21	8.31	-72.67	-46.48	99.93	178.44	350.70	-0.02	0.07	0.08	1.56	3.00
80.9	5.00	180.55	9.00	351.60	10.81	-46.74	12.00	51.84	9.75	75.34	8.25	-72.61	-46.98	99.93	179.00	351.60	-0.02	0.07	0.07	1.00	3.00
81	5.00	181.54	9.00	352.50	10.81	-46.72	12.00	51.87	9.76	75.36	8.25	-72.60	-47.48	99.93	180.55	352.50	-0.02	0.07	0.07	-0.55	3.00
81.1	5.00	180.98	9.00	353.40	10.91	-47.10	11.91	51.42	9.64	75.07	8.37	-72.74	-47.98	99.92	181.54	353.40	-0.02	0.08	0.09	-1.54	3.00
81.2	5.00	179.43	9.00	354.30	11.09	-47.76	11.76	50.69	9.44	74.60	8.58	-73.01	-48.48	99.91	180.98	354.30	-0.02	0.09	0.09	-0.98	3.00
81.3	5.00	178.43	9.00	355.20	11.22	-48.28	11.64	50.13	9.28	74.25	8.74	-73.25	-48.98	99.91	179.43	355.20	-0.02	0.09	0.09	0.57	3.00
81.4	5.00	178.99	9.00	356.10	11.24	-48.37	11.62	50.03	9.26	74.19	8.77	-73.30	-49.48	99.93	178.43	356.10	-0.02	0.07	0.08	1.57	3.00
81.5	5.00	180.55	9.00	357.00	11.19	-48.19	11.66	50.22	9.31	74.31	8.72	-73.21	-49.98	99.94	178.99	357.00	-0.02	0.06	0.07	1.01	3.00
81.6	5.00	181.55	9.00	357.90	11.19	-48.16	11.67	50.25	9.32	74.33	8.71	-73.20	-50.48	99.93	180.55	357.90	-0.02	0.07	0.07	-0.55	-177.90
81.7	5.00	180.99	-533.70	358.80	532.53	-74.58	532.16	74.57	534.91	-73.53	535.28	73.55	-50.98	99.92	181.55	358.80	-0.02	0.08	0.09	-1.55	-358.80
81.8	5.00	179.43	-1076.40	305.43	1079.49	-74.26	1071.70	74.15	1073.33	-73.84	1081.10	73.96	-51.48	99.91	180.99	305.43	-0.02	0.09	0.09	-0.99	-305.43
81.9	5.00	178.43	-916.29	197.79	919.19	-73.80	916.01	73.74	913.41	-74.31	916.60	74.37	-51.98	99.91	179.43	197.79	-0.02	0.09	0.09	0.57	-197.79
82	5.00	178.98	-593.37	106.16	589.19	-73.79	598.37	74.04	597.56	-74.32	588.37	74.07	-52.48	99.93	178.43	106.16	-0.02	0.07	0.08	1.57	-106.16
82.1	5.00	180.55	-318.48	46.82	314.07	-74.48	321.04	74.82	322.92	-73.64	315.99	73.27	-52.98	99.94	178.98	46.82	-0.02	0.06	0.07	1.02	-46.82
82.2	5.00	181.56	-140.47	14.98	138.09	-75.86	140.34	76.09	142.99	-72.31	140.78	72.02	-53.48	99.93	180.55	14.98	-0.02	0.07	0.07	-0.55	-14.98
82.3	5.00	181.00	-44.93	0.93	43.82	-80.35	43.81	80.35	46.54	-68.12	46.56	68.13	-53.98	99.92	181.56	0.93	-0.02	0.08	0.09	-1.56	-0.93
82.4	5.00	179.43	-2.79	-3.56	5.15	34.81	4.87	-29.76	6.25	-22.77	6.47	27.04	-54.48	99.91	181.00	-3.56	-0.02	0.09	0.09	-1.00	3.56
82.5	5.00	178.42	10.69	-3.84	12.83	-51.81	13.14	52.87	10.68	78.89	10.29	-78.46	-54.98	99.91	179.43	-3.84	-0.02	0.09	0.09	0.57	3.84
82.6	5.00	178.98	11.53	-2.77	13.64	-53.25	13.89	54.00	11.39	80.75	11.08	-80.49	-55.47	99.93	178.42	-2.77	-0.03	0.07	0.08	1.58	2.77
82.7	5.00	180.55	8.32	-1.62	10.68	-47.01	10.96	48.36	8.63	71.69	8.27	-70.86	-55.97	99.94	178.98	-1.62	-0.03	0.06	0.07	1.02	1.62
82.8	5.00	181.56	4.86	-0.79	7.75	-35.22	7.99	37.62	6.10	53.13	5.78	-50.69	-56.47	99.93	180.55	-0.79	-0.03	0.07	0.07	-0.55	0.79
82.9	5.00	181.01	2.37	-0.30	6.05	-20.93	6.13	22.93	4.96	28.78	4.86	-26.42	-56.97	99.92	181.56	-0.30	-0.03	0.08	0.09	-1.56	0.30
83	5.00	179.43	0.91	-0.07	5.33	-9.90	5.31	8.97	4.82	9.90	4.84	-10.91	-57.47	99.91	181.01	-0.07	-0.03	0.09	0.09	-1.01	0.07

83.1	5.00	178.41	0.20	0.02	5.06	-3.74	5.05	0.56	4.94	0.57	4.95	-3.83	-57.97	99.91	179.43	0.02	-0.03	0.09	0.09	0.57	-0.02
83.2	5.00	178.97	-0.07	0.04	4.98	-0.26	4.98	-1.91	5.02	-1.89	5.02	-0.25	-58.47	99.93	178.41	0.04	-0.03	0.07	0.08	1.59	-0.04
83.3	5.00	180.54	-0.13	0.04	4.97	2.00	4.96	-0.98	5.04	-0.96	5.04	1.97	-58.97	99.94	178.97	0.04	-0.03	0.06	0.07	1.03	-0.04
83.4	5.00	181.57	-0.11	0.02	4.97	2.79	4.97	0.32	5.03	0.32	5.03	2.76	-59.47	99.93	180.54	0.02	-0.03	0.07	0.07	-0.54	-0.02
83.5	5.00	181.02	-0.07	0.01	4.98	1.80	4.98	0.22	5.02	0.22	5.02	1.78	-59.97	99.92	181.57	0.01	-0.03	0.08	0.09	-1.57	-0.01
83.6	5.00	179.44	-0.04	0.01	4.99	-0.15	4.99	-0.99	5.01	-0.98	5.01	-0.15	-60.47	99.91	181.02	0.01	-0.03	0.09	0.09	-1.02	-0.01
83.7	5.00	178.41	-0.02	0.00	5.00	-1.42	5.00	-1.78	5.00	-1.78	5.00	-1.41	-60.97	99.91	179.44	0.00	-0.03	0.09	0.09	0.56	0.00
83.8	5.00	178.96	-0.01	0.00	5.00	-0.99	5.00	-1.10	5.00	-1.10	5.00	-0.99	-61.47	99.93	178.41	0.00	-0.03	0.07	0.08	1.59	0.00
83.9	5.00	180.54	0.00	0.00	5.00	0.55	5.00	0.54	5.00	0.54	5.00	0.55	-61.97	99.94	178.96	0.00	-0.03	0.06	0.07	1.04	0.00
84	5.00	181.58	0.00	0.00	5.00	1.56	5.00	1.59	5.00	1.59	5.00	1.56	-62.47	99.93	180.54	0.00	-0.03	0.07	0.07	-0.54	0.00
84.1	5.00	181.03	0.00	0.00	5.00	1.01	5.00	1.04	5.00	1.04	5.00	1.01	-62.97	99.92	181.58	0.00	-0.03	0.08	0.08	-1.58	0.00
84.2	5.00	179.44	0.00	0.00	5.00	-0.57	5.00	-0.55	5.00	-0.55	5.00	-0.57	-63.47	99.91	181.03	0.00	-0.03	0.09	0.09	-1.03	0.00
84.3	5.00	178.40	0.00	0.00	5.00	-1.61	5.00	-1.60	5.00	-1.60	5.00	-1.61	-63.97	99.92	179.44	0.00	-0.03	0.08	0.09	0.56	0.00
84.4	5.00	178.95	0.00	0.00	5.00	-1.05	5.00	-1.05	5.00	-1.05	5.00	-1.05	-64.47	99.93	178.40	0.00	-0.03	0.07	0.08	1.60	0.00
84.5	5.00	180.54	0.00	0.00	5.00	0.54	5.00	0.54	5.00	0.54	5.00	0.54	-64.97	99.94	178.95	0.00	-0.03	0.06	0.07	1.05	0.00
84.6	5.00	181.59	0.00	0.00	5.00	1.59	5.00	1.59	5.00	1.59	5.00	1.59	-65.47	99.93	180.54	0.00	-0.03	0.07	0.07	-0.54	0.00
84.7	5.00	181.04	0.00	0.00	5.00	1.04	5.00	1.04	5.00	1.04	5.00	1.04	-65.97	99.92	181.59	0.00	-0.03	0.08	0.08	-1.59	0.00
84.8	5.00	179.44	0.00	0.00	5.00	-0.56	5.00	-0.56	5.00	-0.56	5.00	-0.56	-66.47	99.91	181.04	0.00	-0.03	0.09	0.09	-1.04	0.00
84.9	5.00	178.39	0.00	0.00	5.00	-1.61	5.00	-1.61	5.00	-1.61	5.00	-1.61	-66.97	99.92	179.44	0.00	-0.03	0.08	0.09	0.56	0.00
85	5.00	178.94	0.00	0.00	5.00	-1.06	5.00	-1.06	5.00	-1.06	5.00	-1.06	-67.47	99.93	178.39	0.00	-0.03	0.07	0.08	1.61	0.00
85.1	5.00	180.54	0.00	0.00	5.00	0.54	5.00	0.54	5.00	0.54	5.00	0.54	-67.97	99.94	178.94	0.00	-0.03	0.06	0.07	1.06	0.00
85.2	5.00	181.59	0.00	0.00	5.00	1.59	5.00	1.59	5.00	1.59	5.00	1.59	-68.47	99.93	180.54	0.00	-0.03	0.07	0.07	-0.54	0.00
85.3	5.00	181.05	0.00	0.00	5.00	1.05	5.00	1.05	5.00	1.05	5.00	1.05	-68.97	99.92	181.59	0.00	-0.03	0.08	0.08	-1.59	0.00
85.4	5.00	179.44	0.00	0.00	5.00	-0.56	5.00	-0.56	5.00	-0.56	5.00	-0.56	-69.47	99.91	181.05	0.00	-0.03	0.09	0.09	-1.05	0.00
85.5	5.00	178.38	0.00	0.00	5.00	-1.62	5.00	-1.62	5.00	-1.62	5.00	-1.62	-69.97	99.92	179.44	0.00	-0.03	0.08	0.09	0.56	0.00
85.6	5.00	178.93	0.00	0.00	5.00	-1.07	5.00	-1.07	5.00	-1.07	5.00	-1.07	-70.47	99.93	178.38	0.00	-0.03	0.07	0.08	1.62	0.00
85.7	5.00	180.54	0.00	0.00	5.00	0.54	5.00	0.54	5.00	0.54	5.00	0.54	-70.97	99.94	178.93	0.00	-0.03	0.06	0.07	1.07	0.00
85.8	5.00	181.60	0.00	0.00	5.00	1.60	5.00	1.60	5.00	1.60	5.00	1.60	-71.47	99.93	180.54	0.00	-0.03	0.07	0.07	-0.54	0.00
85.9	5.00	181.06	0.00	0.00	5.00	1.06	5.00	1.06	5.00	1.06	5.00	1.06	-71.97	99.92	181.60	0.00	-0.03	0.08	0.08	-1.60	0.00
86	5.00	179.44	0.00	0.00	5.00	-0.56	5.00	-0.56	5.00	-0.56	5.00	-0.56	-72.47	99.91	181.06	0.00	-0.03	0.09	0.09	-1.06	0.00
86.1	5.00	178.38	0.00	0.00	5.00	-1.62	5.00	-1.62	5.00	-1.62	5.00	-1.62	-72.97	99.92	179.44	0.00	-0.03	0.08	0.09	0.56	0.00
86.2	5.00	178.92	0.00	0.00	5.00	-1.08	5.00	-1.08	5.00	-1.08	5.00	-1.08	-73.47	99.93	178.38	0.00	-0.03	0.07	0.07	1.62	0.00
86.3	5.00	180.54	0.00	0.00	5.00	0.54	5.00	0.54	5.00	0.54	5.00	0.54	-73.97	99.94	178.92	0.00	-0.03	0.06	0.07	1.08	0.00
86.4	5.00	181.61	0.00	0.00	5.00	1.61	5.00	1.61	5.00	1.61	5.00	1.61	-74.47	99.94	180.54	0.00	-0.03	0.06	0.07	-0.54	0.00
86.5	5.00	181.06	0.00	0.00	5.00	1.06	5.00	1.06	5.00	1.06	5.00	1.06	-74.97	99.92	181.61	0.00	-0.03	0.08	0.08	-1.61	0.00
86.6	5.00	179.45	0.00	0.00	5.00	-0.55	5.00	-0.55	5.00	-0.55	5.00	-0.55	-75.47	99.91	181.06	0.00	-0.03	0.09	0.09	-1.06	0.00
86.7	5.00	178.37	0.00	0.00	5.00	-1.63	5.00	-1.63	5.00	-1.63	5.00	-1.63	-75.97	99.92	179.45	0.00	-0.03	0.08	0.09	0.55	0.00
86.8	5.00	178.91	0.00	0.00	5.00	-1.09	5.00	-1.09	5.00	-1.09	5.00	-1.09	-76.47	99.93	178.37	0.00	-0.03	0.07	0.07	1.63	0.00
86.9	5.00	180.54	0.00	0.00	5.00	0.54	5.00	0.54	5.00	0.54	5.00	0.54	-76.97	99.94	178.91	0.00	-0.03	0.06	0.07	1.09	0.00
87	5.00	181.62	0.00	0.00	5.00	1.62	5.00	1.62	5.00	1.62	5.00	1.62	-77.47	99.94	180.54	0.00	-0.03	0.06	0.07	-0.54	0.00
87.1	5.00	181.07	0.00	0.00	5.00	1.07	5.00	1.07	5.00	1.07	5.00	1.07	-77.97	99.92	181.62	0.00	-0.03	0.08	0.08	-1.62	0.00
87.2	5.00	179.45	0.00	0.00	5.00	-0.55	5.00	-0.55	5.00	-0.55	5.00	-0.55	-78.47	99.91	181.07	0.00	-0.03	0.09	0.09	-1.07	0.00
87.3	5.00	178.36	0.00	0.00	5.00	-1.64	5.00	-1.64	5.00	-1.64	5.00	-1.64	-78.97	99.92	179.45	0.00	-0.03	0.08	0.09	0.55	0.00
87.4	5.00	178.90	0.00	0.00	5.00	-1.10	5.00	-1.10	5.00	-1.10	5.00	-1.10	-79.47	99.93	178.36	0.00	-0.03	0.07	0.07	1.64	0.00
87.5	5.00	180.53	0.00	0.00	5.00	0.53	5.00	0.53	5.00	0.53	5.00	0.53	-79.97	99.94	178.90	0.00	-0.03	0.06	0.07	1.10	0.00
87.6	5.00	181.62	0.00	0.00	5.00	1.62	5.00	1.62	5.00	1.62	5.00	1.62	-80.47	99.94	180.53	0.00	-0.03	0.06	0.07	-0.53	0.00
87.7	5.00	181.08	0.00	0.00	5.00	1.08	5.00	1.08	5.00	1.08	5.00	1.08	-80.97	99.92	181.62	0.00	-0.03	0.08	0.08	-1.62	0.00
87.8	5.00	179.45	0.00	0.00	5.00	-0.55	5.00	-0.55	5.00	-0.55	5.00	-0.55	-81.47	99.91	181.08	0.00	-0.03	0.09	0.09	-1.08	0.00
87.9	5.00	178.35	0.00	0.00	5.00	-1.65	5.00	-1.65	5.00	-1.65	5.00	-1.65	-81.97	99.92	179.45	0.00	-0.03	0.08	0.09	0.55	0.00
88	5.00	178.89	0.00	0.00	5.00	-1.11	5.00	-1.11	5.00	-1.11	5.00	-1.11	-82.47	99.93	178.35	0.00	-0.03	0.07	0.07	1.65	0.00
88.1	5.00	180.53	0.00	0.00	5.00	0.53	5.00	0.53	5.00	0.53	5.00	0.53	-82.97	99.94	178.89	0.00	-0.03	0.06	0.07	1.11	0.00
88.2	5.00	181.63	0.00	0.00	5.00	1.63	5.00	1.63	5.00	1.63	5.00	1.63	-83.47	99.94	180.53	0.00	-0.03	0.06	0.07	-0.53	0.00
88.3	5.00	181.09	0.00	0.00	5.00	1.09	5.00	1.09	5.00	1.09	5.00	1.09	-83.97	99.92	181.63	0.00	-0.03	0.08	0.08	-1.63	0.00
88.4	5.00	179.45	0.00	0.00	5.00	-0.55	5.00	-0.55	5.00	-0.55	5.00	-0.55	-84.47	99.91	181.09	0.00	-0.03	0.09	0.09	-1.09	0.00
88.5	5.00	178.35	0.00	0.00	5.00	-1.65	5.00	-1.65	5.00	-1.65	5.00	-1.65	-84.97	99.92	179.45	0.00	-0.03	0.08	0.09	0.55	0.00
88.6	5.00	178.88	0.00	0.00	5.00	-1.12	5.00	-1.12	5.00	-1.12	5.00	-1.12	-85.47	99.93	178.35	0.00	-0.03	0.07	0.07	1.65	0.00
88.7	5.00	180.53	0.00	0.00	5.00	0.53	5.00	0.53	5.00	0.53	5.00	0.53	-85.97	99.94	178.88	0.00	-0.03	0.06	0.07	1.12	0.00
88.8	5.00	181.64	0.00	0.00	5.00	1.64	5.00	1.64	5.00	1.64	5.00	1.64	-86.47	99.94	180.53	0.00	-0.03	0.06	0.07	-0.53	0.00
88.9	5.00	181.10	0.00	0.00	5.00	1.10	5.00	1.10	5.00	1.10	5.00	1.10	-86.97	99.92	181.64	0.00	-0.03	0.08	0.08	-1.64	0.00
89	5.00	179.45	0.00	0.00	5.00	-0.55	5.00	-0.55	5.00	-0.55	5.00	-0.55	-87.47	99.91	181.10	0.00	-0.03	0.09	0.09	-1.10	0.00
89.1	5.00	178.34	0.00	0.00	5.00	-1.66	5.00	-1.66	5.00	-1.66	5.00	-1.66	-87.97	99.92	179.45	0.00	-0.03	0.08	0.09	0.55	0.00
89.2	5.00	178.88	0.00	0.00	5.00	-1.12	5.00	-1.12	5.00	-1.12	5.00	-1.12	-88.47	99.93	178.34	0.00	-0.03	0.07	0.07	1.66	0.00
89.3	5.00	180.53	0.00	0.00	5.00	0.53	5.00	0.53	5.00	0.53	5.00	0.53	-88.97	99.94	178.88	0.00	-0.03	0.06	0.07	1.12	0.00
89.4	5.00	181.65	0.00	0.00	5.00	1.65	5.00	1.65	5.00	1.65	5.00	1.65	-89.47	99.94	180.53	0.00	-0.03	0.06	0.07	-0.53	0.00
89.5	5.00	181.11	0.00	0.00	5.00	1.11	5.00	1.11	5.00	1.11	5.00	1.11	-89.97	99.92	181.65	0.00	-0.03	0.08	0.08	-1.65	0.00
89.6	5.00	179.46	0.00	0.00	5.00	-0.54	5.00	-0.54	5.00	-0.54	5.00	-0.54	-90.47	99.91	181.11	0.00	-0.03	0.09	0.09	-1.11	0.00
89.7	5.00	178.33	0.00	0.00	5.00	-1.67	5.00	-1.67	5.00	-1.67	5.00	-1.67	-90.97	99.92	179.46	0.00	-0.03	0.08	0.09	0.54	0.00
89.8	5.00	178.87	0.00	0.00	5.00	-1.13	5.00	-1.13	5.00	-1.13	5.00	-1.13	-91.47	99.93	178.33	0.00	-0.03	0.07	0.07	1.67	0.00
89.9	5.00	180.53	0.00	0.00	5.00	0.53	5.00	0.53	5.00	0.53	5.00	0.53	-91.97	99.94	178.87	0.00	-0.03	0.06	0.07	1.13	0.00
90	5.00	181.66	0.00	0.00	5.00	1.66	5.00	1.66	5.00	1.66	5.00	1.66	-92.47	99.94	180.53	0.00	-0.03	0.06	0.07	-0.53	0.00
90.1	5.00	181.12	0.00	0.00	5.00	1.12	5.00	1.12	5.00	1.12	5.00	1.12	-92.97	99.92	181.66	0.00	-0.03	0.08	0.08	-1.66	0.00
90.2	5.00	179.46	0.00	0.00	5.00	-0.54	5.00	-0.54	5.00	-0.54	5.00	-0.54	-93.47	99.91	181.12	0.00	-0.03	0.09	0.09	-1.12	0.00
90.3	5.00	178.32	0.00	0.00	5.00	-1.68	5.00	-1.68	5.00	-1.68	5.00	-1.68	-93.97	99.92	179.46	0.00	-0.03	0.08	0.09	0.54	0.00
90.4	5.00	178.86	0.00	0.00	5.00	-1.14	5.00	-1.14	5.00	-1.14	5.00	-1.14	-94.47	99.93	178.32	0.00	-0.03	0.07	0.07	1.68	0.00
90.5	5.00	180.53	0.00	0.00	5.00	0.53	5.00	0.53	5.00	0.53	5.00	0.53	-94.97	99.94	178.86	0.00	-0.03	0.06	0.07	1.14	0.00
90.6	5.00	181.66	0.00	0.00	5.00	1.66	5.00	1.66	5.00	1.66	5.00	1.66	-95.47	99.94	180.53	0.00	-0.03	0.06	0.07	-0.53	0.00
90.7	5.00	181.13	0.00	0.00	5.00	1.13	5.00	1.13	5.00	1.13	5.00	1.13	-95.97	99.92	181.66	0.00	-0.03	0.08	0.08	-1.66	0.00
90.8	5.00	179.46	0.00	0.00	5.00	-0.54	5.00	-0.54	5.00	-0.54	5.00	-0.54	-96.47	99.91	181.13	0.00	-0.03	0.09	0.09	-1.13	0.00
90.9	5.00	178.32	0.00	0.00	5.00	-1.68	5.00	-1.68	5.00	-1.68	5.00	-1.68	-96.97	99.92	179.46	0.00	-0.03	0.08	0.09	0.54	0.00
91	5.00	178.85	0.00	0.00	5.00	-1.15	5.00	-1.15	5.00	-1.15	5.00	-1.15	-97.47	99.93	178.32	0.00	-0.03	0.07	0.07	1.68	0.00
91.1	5.00	180.52	0.00	0.00	5.00	0.52	5.00	0.52	5.00	0.52	5.00	0.52	-97.97	99.94	178.85	0.00	-0.03	0.06	0.07	1.15	0.00
91.2	5.00	181.67	0.00	0.00	5.00	1.67	5.00	1.67	5.00	1.67	5.00	1.67	-98.47	99.94	180.52	0.00	-0.03	0.06	0.07	-0.52	0.00
91.3	5.00	181.14	0.00	0.00	5.00	1.14	5.00	1.14	5.00	1.14	5.00	1.14	-98.97	99.93	181.67	0.00	-0.03	0.07	0.08	-1.67	0.00
91.4	5.00	179.46	0.00	0.00	5.00	-0.54	5.00	-0.54	5.00	-0.54	5.00	-0.54	-99.47	99.92	181.14	0.00	-0.03	0.08	0.09	-1.14	0.00
91.5	5.00	178.31	0.00	0.00	5.00	-1.69	5.00	-1.69	5.00	-1.69	5.00	-1.69	-99.97	99.92	179.46	0.00	-0.03	0.08	0.09	0.54	0.00
91.6	5.00	178.84	0.00	0.00	5.00	-1.16	5.00	-1.16	5.00	-1.16	5.00	-1.16	-100.47	99.93	178.31	0.00	-0.03	0.06	0.07	2.26	0.00
91.7	5.00	181.09	0.00	0.00	5.00	1.09	5.00	1.09	5.00	1.09	5.00	1.09	-100.97	99.94	178.84	0.00	-0.04	0.05	0.06	2.31	0.00
91.8	5.00	183.40	0.00	0.00	5.00	3.40	5.00	3.40	5.00	3.40	5.00	3.40	-101.47	99.94	181.09	0.00	-0.04	0.04	0.06	0.62	0.00
91.9	5.00	184.03	0.00	0.00	5.00	4.03	5.00	4.03	5.00	4.03	5.00	4.03	-101.96	99.91	183.40	0.00	-0.04	0.05	0.07	-1.11	0.00
92	5.00	182.91	0.00	0.00	5.00	2.91	5.00	2.91	5.00	2.91	5.00	2.91	-102.46	99.87	184.03	0.00	-0.04	0.07	0.08	-1.16	0.00
92.1	5.00	181.75	0.00	0.00	5.00	1.75	5.00	1.75	5.00	1.75	5.00	1.75	-102.96	99.84	182.91	0.00	-0.04	0.07	0.07	0.53	0.00
92.2	5.00	182.27	0.00	0.00	5.00	2.27	5.00	2.27	5.00	2.27	5.00	2.27	-103.46	99.83	181.75	0.00	-0.04	0.05	0.06	2.27	0.00
92.3	5.00	184.53	0.00	0.00	5.00	4.53	5.00	4.53	5.00	4.53	5.00	4.53	-103.96	99.81	182.27	0.00	-0.04	0.03	0.05	2.32	0.00
92.4	5.00	186.85	0.00	0.00	5.00	6.85	5.00	6.85	5.00	6.85	5.00	6.85	-104.46	99.77	184.53	0.00	-0.04	0.03	0.04	0.63	0.00
92.5	5.00	187.48	0.00	0.00	5.00	7.48	5.00	7.48	5.00	7.48	5.00	7.48	-104.96	99.71	186.85	0.00	-0.04	0.04	0.05	-1.12	0.00
92.6	5.00	186.36	0.00	0.00	5.00	6.36	5.00	6.36	5.00	6.36	5.00	6.36	-105.45	99.65	187.48	0.00	-0.04	0.05	0.06	-1.17	0.00
92.7	5.00	185.18	0.00	0.00	5.00	5.18	5.00	5.18	5.00	5.18	5.00	5.18	-105.95	99.59	186.36	0.00	-0.04	0.05	0.06	0.52	0.00
92.8	5.00	185.70	0.00	0.00	5.00	5.70	5.00	5.70	5.00	5.70	5.00	5.70	-106.45	99.55	185.18	0.00	-0.04	0.03	0.05	2.27	0.00
92.9	5.00	187.97	0.00	0.00	5.00	7.97	5.00	7.97	5.00	7.97	5.00	7.97	-106.94	99.50	185.70	0.00	-0.04	0.01	0.04	2.33	0.00
93	5.00	190.30	0.00	0.00	5.00	10.30	5.00	10.30	5.00	10.30	5.00	10.30	-107.44	99.43	187.97	0.00	-0.04	0.01	0.04	0.63	0.00
93.1	5.00	190.93	0.00	0.00	5.00	10.93	5.00	10.93	5.00	10.93	5.00	10.93	-107.93	99.34	190.30	0.00	-0.04	0.02	0.04	-1.13	0.00
93.2	5.00	189.81	0.00	0.00	5.00	9.81	5.00	9.81	5.00	9.81	5.00	9.81	-108.42	99.24	190.93	0.00	-0.04	0.04	0.05	-1.19	0.00
93.3	5.00	188.62	0.00	0.00	5.00	8.62	5.00	8.62	5.00	8.62	5.00	8.62	-108.92	99.16	189.81	0.00	-0.04	0.03	0.05	0.51	0.00
93.4	5.00	189.13	0.00	0.00	5.00	9.13	5.00	9.13	5.00	9.13	5.00	9.13	-109.41	99.08	188.62	0.00	-0.04	0.02	0.04	2.27	0.00
93.5	5.00	191.41	0.00	0.00	5.00	11.41	5.00	11.41	5.00	11.41	5.00	11.41	-109.90	99.00	189.13	0.00	-0.03	0.00	0.03	2.33	0.00
93.6	5.00	193.75	0.00	0.00	5.00	13.75	5.00	13.75	5.00	13.75	5.00	13.75	-110.39	98.90	191.41	0.00	-0.03	0.00	0.03	0.63	0.00
93.7	5.00	194.38	0.00	0.00	5.00	14.38	5.00	14.38	5.00	14.38	5.00	14.38	-110.88	98.78	193.75	0.00	-0.04	0.01	0.04	-1.13	0.00
93.8	5.00	193.26	0.00	0.00	5.00	13.26	5.00	13.26	5.00	13.26	5.00	13.26	-111.36	98.66	194.38	0.00	-0.04	0.02	0.05	-1.20	0.00
93.9	5.00	192.06	0.00	0.00	5.00	12.06	5.00	12.06	5.00	12.06	5.00	12.06	-111.85	98.55	193.26	0.00	-0.04	0.02	0.05	0.50	0.00
94	5.00	192.57	0.00	0.00	5.00	12.57	5.00	12.57	5.00	12.57	5.00	12.57	-112.34	98.44	192.06	0.00	-0.04	0.00	0.04	2.27	0.00
94.1	5.00	194.84	0.00	0.00	5.00	14.84	5.00	14.84	5.00	14.84	5.00	14.84	-112.83	98.33	192.57	0.00	-0.03	-0.01	0.04	2.34	0.00
94.2	5.00	197.19	0.00	0.00	5.00	17.19	5.00	17.19	5.00	17.19	5.00	17.19	-113.31	98.20	194.84	0.00	-0.03	-0.02	0.04	0.63	0.00
94.3	5.00	197.84	0.00	0.00	5.00	17.84	5.00	17.84	5.00	17.84	5.00	17.84	-113.79	98.06	197.19	0.00	-0.04	-0.01	0.04	-1.14	0.00
94.4	5.00	196.70	0.00	0.00	5.00	16.70	5.00	16.70	5.00	16.70	5.00	16.70	-114.26	97.90	197.84	0.00	-0.04	0.01	0.04	-1.21	0.00
94.5	5.00	195.50	0.00	0.00	5.00	15.50	5.00	15.50	5.00	15.50	5.00	15.50	-114.74	97.76	196.70	0.00	-0.04	0.00	0.04	0.49	0.00
94.6	5.00	196.00	0.00	0.00	5.00	16.00	5.00	16.00	5.00	16.00	5.00	16.00	-115.23	97.63	195.50	0.00	-0.04	-0.01	0.04	2.27	0.00
94.7	5.00	198.28	0.00	0.00	5.00	18.28	5.00	18.28	5.00	18.28	5.00	18.28	-115.71	97.49	196.00	0.00	-0.03	-0.03	0.04	2.34	0.00
94.8	5.00	200.64	0.00	0.00	5.00	20.64	5.00	20.64	5.00	20.64	5.00	20.64	-116.18	97.33	198.28	0.00	-0.03	-0.03	0.04	0.64	0.00
94.9	5.00	201.29	0.00	0.00	5.00	21.29	5.00	21.29	5.00	21.29	5.00	21.29	-116.65	97.16	200.64	0.00	-0.03	-0.02	0.04	-1.15	0.00

95	5.00	200.15	0.00	0.00	5.00	20.15	5.00	20.15	5.00	20.15	5.00	20.15	-117.11	96.97	201.29	0.00	-0.04	-0.01	0.04	-1.22	0.00
95.1	5.00	198.94	0.00	0.00	5.00	18.94	5.00	18.94	5.00	18.94	5.00	18.94	-117.58	96.80	200.15	0.00	-0.04	-0.01	0.04	0.49	0.00
95.2	5.00	199.43	0.00	0.00	5.00	19.43	5.00	19.43	5.00	19.43	5.00	19.43	-118.06	96.64	198.94	0.00	-0.03	-0.03	0.04	2.27	0.00
95.3	5.00	201.72	0.00	0.00	5.00	21.72	5.00	21.72	5.00	21.72	5.00	21.72	-118.53	96.47	199.43	0.00	-0.03	-0.04	0.05	2.35	0.00
95.4	5.00	204.08	0.00	0.00	5.00	24.08	5.00	24.08	5.00	24.08	5.00	24.08	-118.99	96.29	201.72	0.00	-0.03	-0.05	0.05	0.64	0.00
95.5	5.00	204.73	0.00	0.00	5.00	24.73	5.00	24.73	5.00	24.73	5.00	24.73	-119.45	96.08	204.08	0.00	-0.03	-0.03	0.05	-1.15	0.00
95.6	5.00	203.60	0.00	0.00	5.00	23.60	5.00	23.60	5.00	23.60	5.00	23.60	-119.90	95.87	204.73	0.00	-0.04	-0.02	0.04	-1.23	0.00
95.7	5.00	202.38	0.00	0.00	5.00	22.38	5.00	22.38	5.00	22.38	5.00	22.38	-120.36	95.67	203.60	0.00	-0.04	-0.02	0.04	0.48	0.00
95.8	5.00	202.87	0.00	0.00	5.00	22.87	5.00	22.87	5.00	22.87	5.00	22.87	-120.82	95.48	202.38	0.00	-0.03	-0.04	0.05	2.27	0.00
95.9	5.00	205.16	0.00	0.00	5.00	25.16	5.00	25.16	5.00	25.16	5.00	25.16	-121.28	95.29	202.87	0.00	-0.02	-0.06	0.06	2.35	0.00
96	5.00	207.53	0.00	0.00	5.00	27.53	5.00	27.53	5.00	27.53	5.00	27.53	-121.74	95.08	205.16	0.00	-0.02	-0.06	0.06	0.64	0.00
96.1	5.00	208.18	0.00	0.00	5.00	28.18	5.00	28.18	5.00	28.18	5.00	28.18	-122.18	94.85	207.53	0.00	-0.03	-0.05	0.06	-1.16	0.00
96.2	5.00	207.04	0.00	0.00	5.00	27.04	5.00	27.04	5.00	27.04	5.00	27.04	-122.62	94.61	208.18	0.00	-0.03	-0.04	0.05	-1.24	0.00
96.3	5.00	205.82	0.00	0.00	5.00	25.82	5.00	25.82	5.00	25.82	5.00	25.82	-123.07	94.38	207.04	0.00	-0.03	-0.04	0.05	0.48	0.00
96.4	5.00	206.30	0.00	0.00	5.00	26.30	5.00	26.30	5.00	26.30	5.00	26.30	-123.52	94.17	205.82	0.00	-0.03	-0.05	0.06	2.27	0.00
96.5	5.00	208.59	0.00	0.00	5.00	28.59	5.00	28.59	5.00	28.59	5.00	28.59	-123.96	93.94	206.30	0.00	-0.02	-0.07	0.07	2.36	0.00
96.6	5.00	210.97	0.00	0.00	5.00	30.97	5.00	30.97	5.00	30.97	5.00	30.97	-124.40	93.70	208.59	0.00	-0.02	-0.07	0.08	0.64	0.00
96.7	5.00	211.63	0.00	0.00	5.00	31.63	5.00	31.63	5.00	31.63	5.00	31.63	-124.83	93.45	210.97	0.00	-0.02	-0.06	0.07	-1.16	0.00
96.8	5.00	210.48	0.00	0.00	5.00	30.48	5.00	30.48	5.00	30.48	5.00	30.48	-125.26	93.18	211.63	0.00	-0.03	-0.05	0.06	-1.25	0.00
96.9	5.00	209.25	0.00	0.00	5.00	29.25	5.00	29.25	5.00	29.25	5.00	29.25	-125.69	92.93	210.48	0.00	-0.03	-0.05	0.06	0.47	0.00
97	5.00	209.74	0.00	0.00	5.00	29.74	5.00	29.74	5.00	29.74	5.00	29.74	-126.13	92.69	209.25	0.00	-0.02	-0.07	0.07	2.27	0.00
97.1	5.00	212.03	0.00	0.00	5.00	32.03	5.00	32.03	5.00	32.03	5.00	32.03	-126.56	92.44	209.74	0.00	-0.01	-0.08	0.08	2.36	0.00
97.2	5.00	214.41	0.00	0.00	5.00	34.41	5.00	34.41	5.00	34.41	5.00	34.41	-126.98	92.17	212.03	0.00	-0.01	-0.09	0.09	0.64	0.00
97.3	5.00	215.07	0.00	0.00	5.00	35.07	5.00	35.07	5.00	35.07	5.00	35.07	-127.40	91.89	214.41	0.00	-0.02	-0.08	0.08	-1.16	0.00
97.4	5.00	213.93	0.00	0.00	5.00	33.93	5.00	33.93	5.00	33.93	5.00	33.93	-127.81	91.60	215.07	0.00	-0.03	-0.06	0.07	-1.25	0.00
97.5	5.00	212.69	0.00	0.00	5.00	32.69	5.00	32.69	5.00	32.69	5.00	32.69	-128.22	91.32	213.93	0.00	-0.02	-0.07	0.07	0.47	0.00
97.6	5.00	213.18	0.00	0.00	5.00	33.18	5.00	33.18	5.00	33.18	5.00	33.18	-128.64	91.05	212.69	0.00	-0.02	-0.08	0.08	2.27	0.00
97.7	5.00	215.47	0.00	0.00	5.00	35.47	5.00	35.47	5.00	35.47	5.00	35.47	-129.06	90.78	213.18	0.00	-0.01	-0.10	0.10	2.36	0.00
97.8	5.00	217.86	0.00	0.00	5.00	37.86	5.00	37.86	5.00	37.86	5.00	37.86	-129.47	90.49	215.47	0.00	0.00	-0.10	0.10	0.64	0.00
97.9	5.00	218.52	0.00	0.00	5.00	38.52	5.00	38.52	5.00	38.52	5.00	38.52	-129.86	90.18	217.86	0.00	-0.01	-0.09	0.09	-1.17	0.00
98	5.00	217.37	0.00	0.00	5.00	37.37	5.00	37.37	5.00	37.37	5.00	37.37	-130.25	89.87	218.52	0.00	-0.02	-0.08	0.08	-1.26	0.00
98.1	5.00	216.13	0.00	0.00	5.00	36.13	5.00	36.13	5.00	36.13	5.00	36.13	-130.65	89.57	217.37	0.00	-0.02	-0.08	0.08	0.46	0.00
98.2	5.00	216.62	0.00	0.00	5.00	36.62	5.00	36.62	5.00	36.62	5.00	36.62	-131.05	89.27	216.13	0.00	-0.01	-0.09	0.09	2.27	0.00
98.3	5.00	218.91	0.00	0.00	5.00	38.91	5.00	38.91	5.00	38.91	5.00	38.91	-131.46	88.98	216.62	0.00	0.00	-0.11	0.11	2.36	0.00
98.4	5.00	221.30	0.00	0.00	5.00	41.30	5.00	41.30	5.00	41.30	5.00	41.30	-131.84	88.66	218.91	0.00	0.00	-0.11	0.11	0.64	0.00
98.5	5.00	221.96	0.00	0.00	5.00	41.96	5.00	41.96	5.00	41.96	5.00	41.96	-132.22	88.33	221.30	0.00	0.00	-0.10	0.10	-1.17	0.00
98.6	5.00	220.81	0.00	0.00	5.00	40.81	5.00	40.81	5.00	40.81	5.00	40.81	-132.59	88.00	221.96	0.00	-0.01	-0.09	0.09	-1.26	0.00
98.7	5.00	219.57	0.00	0.00	5.00	39.57	5.00	39.57	5.00	39.57	5.00	39.57	-132.97	87.67	220.81	0.00	-0.01	-0.09	0.09	0.46	0.00
98.8	5.00	220.06	0.00	0.00	5.00	40.06	5.00	40.06	5.00	40.06	5.00	40.06	-133.36	87.35	219.57	0.00	0.00	-0.11	0.11	2.27	0.00
98.9	5.00	222.35	0.00	0.00	5.00	42.35	5.00	42.35	5.00	42.35	5.00	42.35	-133.74	87.03	220.06	0.00	0.01	-0.12	0.12	2.36	0.00
99	5.00	224.74	0.00	0.00	5.00	44.74	5.00	44.74	5.00	44.74	5.00	44.74	-134.11	86.69	222.35	0.00	0.01	-0.12	0.12	0.64	0.00
99.1	5.00	225.40	0.00	0.00	5.00	45.40	5.00	45.40	5.00	45.40	5.00	45.40	-134.46	86.34	224.74	0.00	0.00	-0.11	0.11	-1.17	0.00
99.2	5.00	224.25	0.00	0.00	5.00	44.25	5.00	44.25	5.00	44.25	5.00	44.25	-134.81	85.99	225.40	0.00	-0.01	-0.10	0.10	-1.26	0.00
99.3	5.00	223.01	0.00	0.00	5.00	43.01	5.00	43.01	5.00	43.01	5.00	43.01	-135.17	85.64	224.25	0.00	-0.01	-0.10	0.11	0.46	0.00
99.4	5.00	223.50	0.00	0.00	5.00	43.50	5.00	43.50	5.00	43.50	5.00	43.50	-135.54	85.30	223.01	0.00	0.01	-0.12	0.12	2.27	0.00
99.5	5.00	225.79	0.00	0.00	5.00	45.79	5.00	45.79	5.00	45.79	5.00	45.79	-135.90	84.95	223.50	0.00	0.02	-0.13	0.13	2.36	0.00
99.6	5.00	228.18	0.00	0.00	5.00	48.18	5.00	48.18	5.00	48.18	5.00	48.18	-136.25	84.59	225.79	0.00	0.02	-0.13	0.13	0.64	0.00
99.7	5.00	228.84	0.00	0.00	5.00	48.84	5.00	48.84	5.00	48.84	5.00	48.84	-136.58	84.22	228.18	0.00	0.01	-0.12	0.13	-1.17	0.00
99.8	5.00	227.69	0.00	0.00	5.00	47.69	5.00	47.69	5.00	47.69	5.00	47.69	-136.91	83.84	228.84	0.00	0.00	-0.12	0.12	-1.26	0.00
99.9	5.00	226.45	0.00	0.00	5.00	46.45	5.00	46.45	5.00	46.45	5.00	46.45	-137.25	83.47	227.69	0.00	0.00	-0.12	0.12	0.46	0.00
100	5.00	226.94	0.00	0.00	5.00	46.94	5.00	46.94	5.00	46.94	5.00	46.94	-137.59	83.11	226.45	0.00	0.01	-0.13	0.13	2.27	0.00
100.1	5.00	229.23	0.00	0.00	5.00	49.23	5.00	49.23	5.00	49.23	5.00	49.23	-137.93	82.75	226.94	0.00	0.03	-0.14	0.14	2.36	0.00
100.2	5.00	231.62	0.00	0.00	5.00	51.62	5.00	51.62	5.00	51.62	5.00	51.62	-138.26	82.37	229.23	0.00	0.03	-0.14	0.15	0.64	0.00
100.3	5.00	232.28	0.00	0.00	5.00	52.28	5.00	52.28	5.00	52.28	5.00	52.28	-138.57	81.98	231.62	0.00	0.02	-0.14	0.14	-1.17	0.00
100.4	5.00	231.13	0.00	0.00	5.00	51.13	5.00	51.13	5.00	51.13	5.00	51.13	-138.88	81.58	232.28	0.00	0.01	-0.13	0.13	-1.26	0.00
100.5	5.00	229.89	0.00	0.00	5.00	49.89	5.00	49.89	5.00	49.89	5.00	49.89	-139.19	81.19	231.13	0.00	0.01	-0.13	0.13	0.46	0.00
100.6	5.00	230.38	0.00	0.00	5.00	50.38	5.00	50.38	5.00	50.38	5.00	50.38	-139.51	80.81	229.89	0.00	0.02	-0.14	0.14	2.27	0.00
100.7	5.00	232.67	0.00	0.00	5.00	52.67	5.00	52.67	5.00	52.67	5.00	52.67	-139.83	80.42	230.38	0.00	0.04	-0.15	0.16	2.36	0.00
100.8	5.00	235.06	0.00	0.00	5.00	55.06	5.00	55.06	5.00	55.06	5.00	55.06	-140.13	80.03	232.67	0.00	0.04	-0.15	0.16	0.64	0.00
100.9	5.00	235.72	0.00	0.00	5.00	55.72	5.00	55.72	5.00	55.72	5.00	55.72	-140.42	79.62	235.06	0.00	0.03	-0.15	0.15	-1.17	0.00
101	5.00	234.57	0.00	0.00	5.00	54.57	5.00	54.57	5.00	54.57	5.00	54.57	-140.70	79.20	235.72	0.00	0.02	-0.14	0.14	-1.26	0.00
101.1	5.00	233.34	0.00	0.00	5.00	53.34	5.00	53.34	5.00	53.34	5.00	53.34	-140.99	78.80	234.57	0.00	0.02	-0.14	0.14	0.46	0.00
101.2	5.00	233.82	0.00	0.00	5.00	53.82	5.00	53.82	5.00	53.82	5.00	53.82	-141.29	78.39	233.34	0.00	0.03	-0.15	0.15	2.27	0.00
101.3	5.00	236.11	0.00	0.00	5.00	56.11	5.00	56.11	5.00	56.11	5.00	56.11	-141.59	77.99	233.82	0.00	0.05	-0.16	0.17	2.36	0.00
101.4	5.00	238.49	0.00	0.00	5.00	58.49	5.00	58.49	5.00	58.49	5.00	58.49	-141.86	77.58	236.11	0.00	0.05	-0.16	0.17	0.64	0.00
101.5	5.00	239.16	0.00	0.00	5.00	59.16	5.00	59.16	5.00	59.16	5.00	59.16	-142.13	77.15	238.49	0.00	0.04	-0.16	0.16	-1.17	0.00
101.6	5.00	238.01	0.00	0.00	5.00	58.01	5.00	58.01	5.00	58.01	5.00	58.01	-142.38	76.72	239.16	0.00	0.03	-0.15	0.15	-1.26	0.00
101.7	5.00	236.78	0.00	0.00	5.00	56.78	5.00	56.78	5.00	56.78	5.00	56.78	-142.65	76.30	238.01	0.00	0.03	-0.15	0.15	0.46	0.00
101.8	5.00	237.26	0.00	0.00	5.00	57.26	5.00	57.26	5.00	57.26	5.00	57.26	-142.92	75.88	236.78	0.00	0.04	-0.16	0.17	2.27	0.00
101.9	5.00	239.55	0.00	0.00	5.00	59.55	5.00	59.55	5.00	59.55	5.00	59.55	-143.19	75.46	237.26	0.00	0.06	-0.17	0.18	2.35	0.00
102	5.00	241.93	0.00	0.00	5.00	61.93	5.00	61.93	5.00	61.93	5.00	61.93	-143.45	75.03	239.55	0.00	0.06	-0.17	0.18	0.64	0.00
102.1	5.00	242.59	0.00	0.00	5.00	62.59	5.00	62.59	5.00	62.59	5.00	62.59	-143.68	74.59	241.93	0.00	0.05	-0.16	0.17	-1.17	0.00
102.2	5.00	241.45	0.00	0.00	5.00	61.45	5.00	61.45	5.00	61.45	5.00	61.45	-143.91	74.14	242.59	0.00	0.04	-0.16	0.16	-1.25	0.00
102.3	5.00	240.22	0.00	0.00	5.00	60.22	5.00	60.22	5.00	60.22	5.00	60.22	-144.15	73.70	241.45	0.00	0.04	-0.16	0.16	0.46	0.00
102.4	5.00	240.71	0.00	0.00	5.00	60.71	5.00	60.71	5.00	60.71	5.00	60.71	-144.40	73.27	240.22	0.00	0.05	-0.17	0.18	2.27	0.00
102.5	5.00	243.00	0.00	0.00	5.00	63.00	5.00	63.00	5.00	63.00	5.00	63.00	-144.64	72.83	240.71	0.00	0.07	-0.18	0.19	2.35	0.00
102.6	5.00	245.37	0.00	0.00	5.00	65.37	5.00	65.37	5.00	65.37	5.00	65.37	-144.87	72.39	243.00	0.00	0.07	-0.18	0.19	0.64	0.00
102.7	5.00	246.02	0.00	0.00	5.00	66.02	5.00	66.02	5.00	66.02	5.00	66.02	-145.08	71.93	245.37	0.00	0.06	-0.17	0.18	-1.16	0.00
102.8	5.00	244.88	0.00	0.00	5.00	64.88	5.00	64.88	5.00	64.88	5.00	64.88	-145.28	71.48	246.02	0.00	0.05	-0.17	0.17	-1.25	0.00
102.9	5.00	243.66	0.00	0.00	5.00	63.66	5.00	63.66	5.00	63.66	5.00	63.66	-145.49	71.02	244.88	0.00	0.05	-0.17	0.18	0.47	0.00
103	5.00	244.15	0.00	0.00	5.00	64.15	5.00	64.15	5.00	64.15	5.00	64.15	-145.72	70.57	243.66	0.00	0.07	-0.18	0.19	2.26	0.00
103.1	5.00	246.44	0.00	0.00	5.00	66.44	5.00	66.44	5.00	66.44	5.00	66.44	-145.93	70.12	244.15	0.00	0.08	-0.18	0.20	2.35	0.00
103.2	5.00	248.80	0.00	0.00	5.00	68.80	5.00	68.80	5.00	68.80	5.00	68.80	-146.13	69.67	246.44	0.00	0.08	-0.19	0.20	0.63	0.00
103.3	5.00	249.46	0.00	0.00	5.00	69.46	5.00	69.46	5.00	69.46	5.00	69.46	-146.31	69.20	248.80	0.00	0.07	-0.18	0.20	-1.16	0.00
103.4	5.00	248.32	0.00	0.00	5.00	68.32	5.00	68.32	5.00	68.32	5.00	68.32	-146.49	68.73	249.46	0.00	0.06	-0.18	0.19	-1.24	0.00
103.5	5.00	247.10	0.00	0.00	5.00	67.10	5.00	67.10	5.00	67.10	5.00	67.10	-146.67	68.27	248.32	0.00	0.06	-0.18	0.19	0.47	0.00
103.6	5.00	247.60	0.00	0.00	5.00	67.60	5.00	67.60	5.00	67.60	5.00	67.60	-146.87	67.81	247.10	0.00	0.08	-0.18	0.20	2.26	0.00
103.7	5.00	249.88	0.00	0.00	5.00	69.88	5.00	69.88	5.00	69.88	5.00	69.88	-147.06	67.34	247.60	0.00	0.09	-0.19	0.21	2.34	0.00
103.8	5.00	252.24	0.00	0.00	5.00	72.24	5.00	72.24	5.00	72.24	5.00	72.24	-147.23	66.87	249.88	0.00	0.10	-0.19	0.22	0.63	0.00
103.9	5.00	252.89	0.00	0.00	5.00	72.89	5.00	72.89	5.00	72.89	5.00	72.89	-147.38	66.40	252.24	0.00	0.08	-0.19	0.21	-1.16	0.00
104	5.00	251.75	0.00	0.00	5.00	71.75	5.00	71.75	5.00	71.75	5.00	71.75	-147.53	65.92	252.89	0.00	0.07	-0.18	0.20	-1.23	0.00
104.1	5.00	250.54	0.00	0.00	5.00	70.54	5.00	70.54	5.00	70.54	5.00	70.54	-147.69	65.45	251.75	0.00	0.07	-0.19	0.20	0.48	0.00
104.2	5.00	251.04	0.00	0.00	5.00	71.04	5.00	71.04	5.00	71.04	5.00	71.04	-147.85	64.97	250.54	0.00	0.09	-0.19	0.21	2.26	0.00
104.3	5.00	253.32	0.00	0.00	5.00	73.32	5.00	73.32	5.00	73.32	5.00	73.32	-148.02	64.50	251.04	0.00	0.11	-0.20	0.22	2.33	0.00
104.4	5.00	255.67	0.00	0.00	5.00	75.67	5.00	75.67	5.00	75.67	5.00	75.67	-148.16	64.02	253.32	0.00	0.11	-0.20	0.23	0.63	0.00
104.5	5.00	256.32	0.00	0.00	5.00	76.32	5.00	76.32	5.00	76.32	5.00	76.32	-148.28	63.54	255.67	0.00	0.10	-0.19	0.22	-1.15	0.00
104.6	5.00	255.19	0.00	0.00	5.00	75.19	5.00	75.19	5.00	75.19	5.00	75.19	-148.40	63.05	256.32	0.00	0.08	-0.19	0.21	-1.22	0.00
104.7	5.00	253.99	0.00	0.00	5.00	73.99	5.00	73.99	5.00	73.99	5.00	73.99	-148.53	62.57	255.19	0.00	0.09	-0.19	0.21	0.48	0.00
104.8	5.00	254.49	0.00	0.00	5.00	74.49	5.00	74.49	5.00	74.49	5.00	74.49	-148.67	62.09	253.99	0.00	0.10	-0.20	0.22	2.26	0.00
104.9	5.00	256.76	0.00	0.00	5.00	76.76	5.00	76.76	5.00	76.76	5.00	76.76	-148.80	61.61	254.49	0.00	0.12	-0.20	0.23	2.33	0.00
105	5.00	259.11	0.00	0.00	5.00	79.11	5.00	79.11	5.00	79.11	5.00	79.11	-148.92	61.12	256.76	0.00	0.12	-0.20	0.24	0.62	0.00
105.1	5.00	259.75	0.00	0.00	5.00	79.75	5.00	79.75	5.00	79.75	5.00	79.75	-149.01	60.63	259.11	0.00	0.11	-0.20	0.23	-1.15	0.00
105.2	5.00	258.62	0.00	0.00	5.00	78.62	5.00	78.62	5.00	78.62	5.00	78.62	-149.10	60.14	259.75	0.00	0.10	-0.20	0.22	-1.21	0.00
105.3	5.00	257.43	0.00	0.00	5.00	77.43	5.00	77.43	5.00	77.43	5.00	77.43	-149.20	59.65	258.62	0.00	0.10	-0.20	0.22	0.49	0.00
105.4	5.00	257.94	0.00	0.00	5.00	77.94	5.00	77.94	5.00	77.94	5.00	77.94	-149.31	59.16	257.43	0.00	0.12	-0.20	0.23	2.25	0.00
105.5	5.00	260.21	0.00	0.00	5.00	80.21	5.00	80.21	5.00	80.21	5.00	80.21	-149.41	58.67	257.94	0.00	0.13	-0.21	0.25	2.32	0.00
105.6	5.00	262.54	0.00	0.00	5.00	82.54	5.00	82.54	5.00	82.54	5.00	82.54	-149.50	58.18	260.21	0.00	0.14	-0.21	0.25	0.62	0.00
105.7	5.00	263.18	0.00	0.00	5.00	83.18	5.00	83.18	5.00	83.18	5.00	83.18	-149.56	57.68	262.54	0.00	0.12	-0.21	0.24	-1.14	0.00
105.8	5.00	262.05	0.00	0.00	5.00	82.05	5.00	82.05	5.00	82.05	5.00	82.05	-149.62	57.18	263.18	0.00	0.11	-0.20	0.23	-1.20	0.00
105.9	5.00	260.87	0.00	0.00	5.00	80.87	5.00	80.87	5.00	80.87	5.00	80.87	-149.69	56.69	262.05	0.00	0.11	-0.20	0.23	0.49	0.00
106	5.00	261.38	0.00	0.00	5.00	81.38	5.00	81.38	5.00	81.38	5.00	81.38	-149.77	56.20	260.87	0.00	0.13	-0.21	0.24	2.25	0.00
106.1	5.00	263.65	0.00	0.00	5.00	83.65	5.00	83.65	5.00	83.65	5.00	83.65	-149.84	55.70	261.38	0.00	0.15	-0.21	0.26	2.31	0.00
106.2	5.00	265.97	0.00	0.00	5.00	85.97	5.00	85.97	5.00	85.97	5.00	85.97	-149.90	55.20	263.65	0.00	0.15	-0.21	0.26	0.62	0.00
106.3	5.00	266.60	0.00	0.00	5.00	86.60	5.00	86.60	5.00	86.60	5.00	86.60	-149.93	54.71	265.97	0.00	0.14	-0.21	0.25	-1.13	0.00
106.4	5.00	265.49	0.00	0.00	5.00	85.49	5.00	85.49	5.00	85.49	5.00	85.49	-149.96	54.21	266.60	0.00	0.12	-0.21	0.24	-1.19	0.00
106.5	5.00	264.31	0.00	0.00	5.00	84.31	5.00	84.31	5.00	84.31	5.00	84.31	-150.00	53.71	265.49	0.00	0.13	-0.21	0.24	0.50	0.00
106.6	5.00	264.83	0.00	0.00	5.00	84.83	5.00	84.83	5.00	84.83	5.00	84.83	-150.05	53.21	264.31	0.00	0.14	-0.21	0.25	2.25	0.00
106.7	5.00	267.09	0.00	0.00	5.00	87.09	5.00	87.09	5.00	87.09	5.00	87.09	-150.10	52.71	264.83	0.00	0.16	-0.21	0.27	2.30	0.00
106.8	5.00	269.40	0.00	0.00	5.00	89.40	5.00	89.40	5.00	89.40	5.00	89.40	-150.12	52.21	267.09	0.00	0.16	-0.21	0.27	0.61	0.00

106.9	5.00	270.03	0.00	0.00	5.00	-89.97	5.00	-89.97	5.00	-89.97	5.00	-89.97	-150.13	51.71	269.40	0.00	0.15	-0.21	0.26	-1.12	0.00
107	5.00	268.92	0.00	0.00	5.00	88.92	5.00	88.92	5.00	88.92	5.00	88.92	-150.13	51.21	270.03	0.00	0.14	-0.21	0.25	-1.18	0.00
107.1	5.00	267.76	0.00	0.00	5.00	87.76	5.00	87.76	5.00	87.76	5.00	87.76	-150.14	50.71	268.92	0.00	0.14	-0.21	0.26	0.51	0.00
107.2	5.00	268.28	0.00	0.00	5.00	88.28	5.00	88.28	5.00	88.28	5.00	88.28	-150.16	50.21	267.76	0.00	0.16	-0.21	0.27	2.24	0.00
107.3	5.00	270.54	0.00	0.00	5.00	-89.46	5.00	-89.46	5.00	-89.46	5.00	-89.46	-150.17	49.71	268.28	0.00	0.17	-0.21	0.28	2.29	0.00
107.4	5.00	272.84	0.00	0.00	5.00	-87.16	5.00	-87.16	5.00	-87.16	5.00	-87.16	-150.17	49.21	270.54	0.00	0.18	-0.21	0.28	0.61	0.00
107.5	5.00	273.45	0.00	0.00	5.00	-86.55	5.00	-86.55	5.00	-86.55	5.00	-86.55	-150.14	48.71	272.84	0.00	0.17	-0.22	0.27	-1.12	0.00
107.6	5.00	272.35	0.00	0.00	5.00	-87.65	5.00	-87.65	5.00	-87.65	5.00	-87.65	-150.11	48.22	273.45	0.00	0.15	-0.22	0.26	-1.16	0.00
107.7	5.00	271.20	0.00	0.00	5.00	-88.80	5.00	-88.80	5.00	-88.80	5.00	-88.80	-150.09	47.72	272.35	0.00	0.15	-0.22	0.27	0.52	0.00
107.8	5.00	271.73	0.00	0.00	5.00	-88.27	5.00	-88.27	5.00	-88.27	5.00	-88.27	-150.08	47.22	271.20	0.00	0.17	-0.22	0.28	2.24	0.00
107.9	5.00	273.98	0.00	0.00	5.00	-86.02	5.00	-86.02	5.00	-86.02	5.00	-86.02	-150.07	46.72	271.73	0.00	0.19	-0.22	0.29	2.28	0.00
108	5.00	276.27	0.00	0.00	5.00	-83.73	5.00	-83.73	5.00	-83.73	5.00	-83.73	-150.03	46.22	273.98	0.00	0.19	-0.22	0.29	0.61	0.00
108.1	5.00	276.88	0.00	0.00	5.00	-83.12	5.00	-83.12	5.00	-83.12	5.00	-83.12	-149.98	45.72	276.27	0.00	0.18	-0.22	0.28	-1.11	0.00
108.2	5.00	275.78	0.00	0.00	5.00	-84.22	5.00	-84.22	5.00	-84.22	5.00	-84.22	-149.92	45.22	276.88	0.00	0.17	-0.22	0.28	-1.15	0.00
108.3	5.00	274.64	0.00	0.00	5.00	-85.36	5.00	-85.36	5.00	-85.36	5.00	-85.36	-149.87	44.73	275.78	0.00	0.17	-0.22	0.28	0.53	0.00
108.4	5.00	275.18	0.00	0.00	5.00	-84.82	5.00	-84.82	5.00	-84.82	5.00	-84.82	-149.83	44.23	274.64	0.00	0.19	-0.22	0.29	2.24	0.00
108.5	5.00	277.42	0.00	0.00	5.00	-82.58	5.00	-82.58	5.00	-82.58	5.00	-82.58	-149.78	43.73	275.18	0.00	0.20	-0.22	0.30	2.27	0.00
108.6	5.00	279.70	0.00	0.00	5.00	-80.30	5.00	-80.30	5.00	-80.30	5.00	-80.30	-149.72	43.23	277.42	0.00	0.21	-0.22	0.30	0.60	0.00
108.7	5.00	280.30	0.00	0.00	5.00	-79.70	5.00	-79.70	5.00	-79.70	5.00	-79.70	-149.63	42.74	279.70	0.00	0.19	-0.22	0.29	-1.10	0.00
108.8	5.00	279.21	0.00	0.00	5.00	-80.79	5.00	-80.79	5.00	-80.79	5.00	-80.79	-149.54	42.25	280.30	0.00	0.18	-0.22	0.29	-1.13	0.00
108.9	5.00	278.09	0.00	0.00	5.00	-81.91	5.00	-81.91	5.00	-81.91	5.00	-81.91	-149.46	41.76	279.21	0.00	0.18	-0.22	0.29	0.54	0.00
109	5.00	278.63	0.00	0.00	5.00	-81.37	5.00	-81.37	5.00	-81.37	5.00	-81.37	-149.39	41.26	278.09	0.00	0.20	-0.22	0.30	2.23	0.00
109.1	5.00	280.87	0.00	0.00	5.00	-79.13	5.00	-79.13	5.00	-79.13	5.00	-79.13	-149.32	40.77	278.63	0.00	0.22	-0.21	0.31	2.26	0.00
109.2	5.00	283.13	0.00	0.00	5.00	-76.87	5.00	-76.87	5.00	-76.87	5.00	-76.87	-149.22	40.28	280.87	0.00	0.22	-0.21	0.31	0.60	0.00
109.3	5.00	283.73	0.00	0.00	5.00	-76.27	5.00	-76.27	5.00	-76.27	5.00	-76.27	-149.11	39.79	283.13	0.00	0.21	-0.22	0.30	-1.09	0.00
109.4	5.00	282.64	0.00	0.00	5.00	-77.36	5.00	-77.36	5.00	-77.36	5.00	-77.36	-148.99	39.30	283.73	0.00	0.20	-0.22	0.30	-1.11	0.00
109.5	5.00	281.53	0.00	0.00	5.00	-78.47	5.00	-78.47	5.00	-78.47	5.00	-78.47	-148.88	38.82	282.64	0.00	0.20	-0.22	0.30	0.55	0.00
109.6	5.00	282.08	0.00	0.00	5.00	-77.92	5.00	-77.92	5.00	-77.92	5.00	-77.92	-148.78	38.33	281.53	0.00	0.22	-0.22	0.31	2.23	0.00
109.7	5.00	284.31	0.00	0.00	5.00	-75.69	5.00	-75.69	5.00	-75.69	5.00	-75.69	-148.68	37.84	282.08	0.00	0.23	-0.21	0.32	2.25	0.00
109.8	5.00	286.56	0.00	0.00	5.00	-73.44	5.00	-73.44	5.00	-73.44	5.00	-73.44	-148.55	37.35	284.31	0.00	0.24	-0.21	0.32	0.60	0.00
109.9	5.00	287.15	0.00	0.00	5.00	-72.85	5.00	-72.85	5.00	-72.85	5.00	-72.85	-148.41	36.87	286.56	0.00	0.22	-0.22	0.31	-1.08	0.00
110	5.00	286.07	0.00	0.00	5.00	-73.93	5.00	-73.93	5.00	-73.93	5.00	-73.93	-148.26	36.39	287.15	0.00	0.21	-0.22	0.31	-1.10	0.00
110.1	5.00	284.97	0.00	0.00	5.00	-75.03	5.00	-75.03	5.00	-75.03	5.00	-75.03	-148.13	35.91	286.07	0.00	0.22	-0.22	0.31	0.56	0.00
110.2	5.00	285.53	0.00	0.00	5.00	-74.47	5.00	-74.47	5.00	-74.47	5.00	-74.47	-148.00	35.43	284.97	0.00	0.23	-0.21	0.32	2.23	0.00
110.3	5.00	287.75	0.00	0.00	5.00	-72.25	5.00	-72.25	5.00	-72.25	5.00	-72.25	-147.86	34.95	285.53	0.00	0.25	-0.21	0.33	2.24	0.00
110.4	5.00	289.99	0.00	0.00	5.00	-70.01	5.00	-70.01	5.00	-70.01	5.00	-70.01	-147.71	34.47	287.75	0.00	0.25	-0.21	0.33	0.59	0.00
110.5	5.00	290.57	0.00	0.00	5.00	-69.43	5.00	-69.43	5.00	-69.43	5.00	-69.43	-147.54	34.00	289.99	0.00	0.24	-0.21	0.32	-1.07	0.00
110.6	5.00	289.50	0.00	0.00	5.00	-70.50	5.00	-70.50	5.00	-70.50	5.00	-70.50	-147.36	33.54	290.57	0.00	0.23	-0.22	0.32	-1.08	0.00
110.7	5.00	288.41	0.00	0.00	5.00	-71.59	5.00	-71.59	5.00	-71.59	5.00	-71.59	-147.20	33.06	289.50	0.00	0.23	-0.22	0.32	0.57	0.00
110.8	5.00	288.98	0.00	0.00	5.00	-71.02	5.00	-71.02	5.00	-71.02	5.00	-71.02	-147.04	32.59	288.41	0.00	0.25	-0.21	0.33	2.22	0.00
110.9	5.00	291.20	0.00	0.00	5.00	-68.80	5.00	-68.80	5.00	-68.80	5.00	-68.80	-146.88	32.12	288.98	0.00	0.26	-0.21	0.33	2.23	0.00
111	5.00	293.41	0.00	0.00	5.00	-66.59	5.00	-66.59	5.00	-66.59	5.00	-66.59	-146.70	31.65	291.20	0.00	0.27	-0.21	0.34	0.59	0.00
111.1	5.00	293.99	0.00	0.00	5.00	-66.01	5.00	-66.01	5.00	-66.01	5.00	-66.01	-146.50	31.19	293.41	0.00	0.26	-0.21	0.33	-1.06	0.00
111.2	5.00	292.92	0.00	0.00	5.00	-67.08	5.00	-67.08	5.00	-67.08	5.00	-67.08	-146.29	30.74	293.99	0.00	0.24	-0.22	0.33	-1.06	0.00
111.3	5.00	291.86	0.00	0.00	5.00	-68.14	5.00	-68.14	5.00	-68.14	5.00	-68.14	-146.10	30.27	292.92	0.00	0.25	-0.21	0.33	0.58	0.00
111.4	5.00	292.43	0.00	0.00	5.00	-67.57	5.00	-67.57	5.00	-67.57	5.00	-67.57	-145.91	29.81	291.86	0.00	0.26	-0.21	0.34	2.22	0.00
111.5	5.00	294.64	0.00	0.00	5.00	-65.36	5.00	-65.36	5.00	-65.36	5.00	-65.36	-145.72	29.35	292.43	0.00	0.28	-0.20	0.34	2.22	0.00
111.6	5.00	296.84	0.00	0.00	5.00	-63.16	5.00	-63.16	5.00	-63.16	5.00	-63.16	-145.51	28.89	294.64	0.00	0.28	-0.20	0.35	0.58	0.00
111.7	5.00	297.41	0.00	0.00	5.00	-62.59	5.00	-62.59	5.00	-62.59	5.00	-62.59	-145.29	28.45	296.84	0.00	0.27	-0.21	0.34	-1.05	0.00
111.8	5.00	296.35	0.00	0.00	5.00	-63.65	5.00	-63.65	5.00	-63.65	5.00	-63.65	-145.06	28.00	297.41	0.00	0.26	-0.21	0.34	-1.04	0.00
111.9	5.00	295.30	0.00	0.00	5.00	-64.70	5.00	-64.70	5.00	-64.70	5.00	-64.70	-144.84	27.56	296.35	0.00	0.26	-0.21	0.34	0.59	0.00
112	5.00	295.88	0.00	0.00	5.00	-64.12	5.00	-64.12	5.00	-64.12	5.00	-64.12	-144.62	27.10	295.30	0.00	0.28	-0.20	0.34	2.21	0.00
112.1	5.00	298.08	0.00	0.00	5.00	-61.92	5.00	-61.92	5.00	-61.92	5.00	-61.92	-144.40	26.65	295.88	0.00	0.29	-0.20	0.35	2.21	0.00
112.2	5.00	300.27	0.00	0.00	5.00	-59.73	5.00	-59.73	5.00	-59.73	5.00	-59.73	-144.17	26.21	298.08	0.00	0.30	-0.20	0.35	0.58	0.00
112.3	5.00	300.83	0.00	0.00	5.00	-59.17	5.00	-59.17	5.00	-59.17	5.00	-59.17	-143.92	25.78	300.27	0.00	0.29	-0.20	0.35	-1.03	0.00
112.4	5.00	299.78	0.00	0.00	5.00	-60.22	5.00	-60.22	5.00	-60.22	5.00	-60.22	-143.66	25.35	300.83	0.00	0.28	-0.21	0.34	-1.02	0.00
112.5	5.00	298.74	0.00	0.00	5.00	-61.26	5.00	-61.26	5.00	-61.26	5.00	-61.26	-143.41	24.92	299.78	0.00	0.28	-0.21	0.35	0.60	0.00
112.6	5.00	299.33	0.00	0.00	5.00	-60.67	5.00	-60.67	5.00	-60.67	5.00	-60.67	-143.17	24.48	298.74	0.00	0.29	-0.20	0.35	2.21	0.00
112.7	5.00	301.52	0.00	0.00	5.00	-58.48	5.00	-58.48	5.00	-58.48	5.00	-58.48	-142.93	24.04	299.33	0.00	0.31	-0.19	0.36	2.20	0.00
112.8	5.00	303.70	0.00	0.00	5.00	-56.30	5.00	-56.30	5.00	-56.30	5.00	-56.30	-142.66	23.62	301.52	0.00	0.31	-0.19	0.36	0.58	0.00
112.9	5.00	304.25	0.00	0.00	5.00	-55.75	5.00	-55.75	5.00	-55.75	5.00	-55.75	-142.39	23.20	303.70	0.00	0.30	-0.19	0.36	-1.02	0.00
113	5.00	303.21	0.00	0.00	5.00	-56.79	5.00	-56.79	5.00	-56.79	5.00	-56.79	-142.11	22.79	304.25	0.00	0.29	-0.20	0.35	-1.00	0.00
113.1	5.00	302.19	0.00	0.00	5.00	-57.81	5.00	-57.81	5.00	-57.81	5.00	-57.81	-141.83	22.37	303.21	0.00	0.29	-0.20	0.36	0.61	0.00
113.2	5.00	302.78	0.00	0.00	5.00	-57.22	5.00	-57.22	5.00	-57.22	5.00	-57.22	-141.57	21.95	302.19	0.00	0.31	-0.19	0.36	2.21	0.00
113.3	5.00	304.97	0.00	0.00	5.00	-55.03	5.00	-55.03	5.00	-55.03	5.00	-55.03	-141.30	21.53	302.78	0.00	0.32	-0.18	0.37	2.19	0.00
113.4	5.00	307.12	0.00	0.00	5.00	-52.88	5.00	-52.88	5.00	-52.88	5.00	-52.88	-141.01	21.12	304.97	0.00	0.32	-0.18	0.37	0.57	0.00
113.5	5.00	307.67	0.00	0.00	5.00	-52.33	5.00	-52.33	5.00	-52.33	5.00	-52.33	-140.71	20.72	307.12	0.00	0.31	-0.19	0.37	-1.01	0.00
113.6	5.00	306.63	0.00	0.00	5.00	-53.37	5.00	-53.37	5.00	-53.37	5.00	-53.37	-140.40	20.32	307.67	0.00	0.31	-0.19	0.36	-0.98	0.00
113.7	5.00	305.63	0.00	0.00	5.00	-54.37	5.00	-54.37	5.00	-54.37	5.00	-54.37	-140.10	19.92	306.63	0.00	0.31	-0.19	0.36	0.63	0.00
113.8	5.00	306.23	0.00	0.00	5.00	-53.77	5.00	-53.77	5.00	-53.77	5.00	-53.77	-139.81	19.51	305.63	0.00	0.32	-0.18	0.37	2.20	0.00
113.9	5.00	308.41	0.00	0.00	5.00	-51.59	5.00	-51.59	5.00	-51.59	5.00	-51.59	-139.52	19.11	306.23	0.00	0.34	-0.17	0.38	2.17	0.00
114	5.00	310.55	0.00	0.00	5.00	-49.45	5.00	-49.45	5.00	-49.45	5.00	-49.45	-139.21	18.72	308.41	0.00	0.34	-0.17	0.38	0.57	0.00
114.1	5.00	311.09	0.00	0.00	5.00	-48.91	5.00	-48.91	5.00	-48.91	5.00	-48.91	-138.88	18.34	310.55	0.00	0.33	-0.18	0.38	-1.00	0.00
114.2	5.00	310.06	0.00	0.00	5.00	-49.94	5.00	-49.94	5.00	-49.94	5.00	-49.94	-138.55	17.96	311.09	0.00	0.32	-0.19	0.37	-0.96	0.00
114.3	5.00	309.07	0.00	0.00	5.00	-50.93	5.00	-50.93	5.00	-50.93	5.00	-50.93	-138.23	17.58	310.06	0.00	0.32	-0.19	0.37	0.64	0.00
114.4	5.00	309.68	0.00	0.00	5.00	-50.32	5.00	-50.32	5.00	-50.32	5.00	-50.32	-137.91	17.19	309.07	0.00	0.34	-0.17	0.38	2.20	0.00
114.5	5.00	311.85	0.00	0.00	5.00	-48.15	5.00	-48.15	5.00	-48.15	5.00	-48.15	-137.60	16.81	309.68	0.00	0.35	-0.16	0.39	2.16	0.00
114.6	5.00	313.98	0.00	0.00	5.00	-46.02	5.00	-46.02	5.00	-46.02	5.00	-46.02	-137.26	16.43	311.85	0.00	0.35	-0.16	0.39	0.57	0.00
114.7	5.00	314.51	0.00	0.00	5.00	-45.49	5.00	-45.49	5.00	-45.49	5.00	-45.49	-136.91	16.07	313.98	0.00	0.34	-0.17	0.38	-0.98	0.00
114.8	5.00	313.49	0.00	0.00	5.00	-46.51	5.00	-46.51	5.00	-46.51	5.00	-46.51	-136.56	15.72	314.51	0.00	0.34	-0.18	0.38	-0.94	0.00
114.9	5.00	312.52	0.00	0.00	5.00	-47.48	5.00	-47.48	5.00	-47.48	5.00	-47.48	-136.22	15.36	313.49	0.00	0.34	-0.18	0.38	0.65	0.00
115	5.00	313.13	0.00	0.00	5.00	-46.87	5.00	-46.87	5.00	-46.87	5.00	-46.87	-135.88	14.99	312.52	0.00	0.35	-0.16	0.39	2.20	0.00
115.1	5.00	315.29	0.00	0.00	5.00	-44.71	5.00	-44.71	5.00	-44.71	5.00	-44.71	-135.54	14.62	313.13	0.00	0.36	-0.15	0.39	2.15	0.00
115.2	5.00	317.41	0.00	0.00	5.00	-42.59	5.00	-42.59	5.00	-42.59	5.00	-42.59	-135.19	14.27	315.29	0.00	0.36	-0.15	0.39	0.57	0.00
115.3	5.00	317.93	0.00	0.00	5.00	-42.07	5.00	-42.07	5.00	-42.07	5.00	-42.07	-134.82	13.93	317.41	0.00	0.36	-0.16	0.39	-0.97	0.00
115.4	5.00	316.92	0.00	0.00	5.00	-43.08	5.00	-43.08	5.00	-43.08	5.00	-43.08	-134.45	13.60	317.93	0.00	0.35	-0.17	0.39	-0.92	0.00
115.5	5.00	315.96	0.00	0.00	5.00	-44.04	5.00	-44.04	5.00	-44.04	5.00	-44.04	-134.08	13.26	316.92	0.00	0.35	-0.17	0.39	0.66	0.00
115.6	5.00	316.58	0.00	0.00	5.00	-43.42	5.00	-43.42	5.00	-43.42	5.00	-43.42	-133.72	12.91	315.96	0.00	0.36	-0.15	0.40	2.19	0.00
115.7	5.00	318.74	0.00	0.00	5.00	-41.26	5.00	-41.26	5.00	-41.26	5.00	-41.26	-133.36	12.56	316.58	0.00	0.38	-0.14	0.40	2.14	0.00
115.8	5.00	320.83	0.00	0.00	5.00	-39.17	5.00	-39.17	5.00	-39.17	5.00	-39.17	-132.98	12.23	318.74	0.00	0.38	-0.14	0.40	0.56	0.00
115.9	5.00	321.35	0.00	0.00	5.00	-38.65	5.00	-38.65	5.00	-38.65	5.00	-38.65	-132.59	11.92	320.83	0.00	0.37	-0.15	0.40	-0.96	0.00
116	5.00	320.34	0.00	0.00	5.00	-39.66	5.00	-39.66	5.00	-39.66	5.00	-39.66	-132.20	11.61	321.35	0.00	0.36	-0.16	0.40	-0.90	0.00
116.1	5.00	319.40	0.00	0.00	5.00	-40.60	5.00	-40.60	5.00	-40.60	5.00	-40.60	-131.82	11.29	320.34	0.00	0.37	-0.15	0.40	0.68	0.00
116.2	5.00	320.03	0.00	0.00	5.00	-39.97	5.00	-39.97	5.00	-39.97	5.00	-39.97	-131.44	10.96	319.40	0.00	0.38	-0.14	0.40	2.19	0.00
116.3	5.00	322.18	0.00	0.00	5.00	-37.82	5.00	-37.82	5.00	-37.82	5.00	-37.82	-131.06	10.64	320.03	0.00	0.39	-0.13	0.41	2.13	0.00
116.4	5.00	324.26	0.00	0.00	5.00	-35.74	5.00	-35.74	5.00	-35.74	5.00	-35.74	-130.66	10.33	322.18	0.00	0.39	-0.13	0.41	0.56	0.00
116.5	5.00	324.77	0.00	0.00	5.00	-35.23	5.00	-35.23	5.00	-35.23	5.00	-35.23	-130.26	10.04	324.26	0.00	0.38	-0.14	0.41	-0.95	0.00
116.6	5.00	323.77	0.00	0.00	5.00	-36.23	5.00	-36.23	5.00	-36.23	5.00	-36.23	-129.85	9.75	324.77	0.00	0.38	-0.15	0.40	-0.88	0.00
116.7	5.00	322.84	0.00	0.00	5.00	-37.16	5.00	-37.16	5.00	-37.16	5.00	-37.16	-129.44	9.46	323.77	0.00	0.38	-0.14	0.41	0.69	0.00
116.8	5.00	323.48	0.00	0.00	5.00	-36.52	5.00	-36.52	5.00	-36.52	5.00	-36.52	-129.05	9.16	322.84	0.00	0.39	-0.13	0.41	2.19	0.00
116.9	5.00	325.62	0.00	0.00	5.00	-34.38	5.00	-34.38	5.00	-34.38	5.00	-34.38	-128.64	8.86	323.48	0.00	0.40	-0.12	0.42	2.12	0.00
117	5.00	327.68	0.00	0.00	5.00	-32.32	5.00	-32.32	5.00	-32.32	5.00	-32.32	-128.23	8.58	325.62	0.00	0.40	-0.11	0.42	0.56	0.00
117.1	5.00	328.19	0.00	0.00	5.00	-31.81	5.00	-31.81	5.00	-31.81	5.00	-31.81	-127.81	8.31	327.68	0.00	0.39	-0.12	0.41	-0.93	0.00
117.2	5.00	327.20	0.00	0.00	5.00	-32.80	5.00	-32.80	5.00	-32.80	5.00	-32.80	-127.38	8.04	328.19	0.00	0.39	-0.13	0.41	-0.86	0.00
117.3	5.00	326.29	0.00	0.00	5.00	-33.71	5.00	-33.71	5.00	-33.71	5.00	-33.71	-126.96	7.77	327.20	0.00	0.39	-0.13	0.41	0.70	0.00
117.4	5.00	326.93	0.00	0.00	5.00	-33.07	5.00	-33.07	5.00	-33.07	5.00	-33.07	-126.55	7.50	326.29	0.00	0.40	-0.12	0.42	2.19	0.00
117.5	5.00	329.06	0.00	0.00	5.00	-30.94	5.00	-30.94	5.00	-30.94	5.00	-30.94	-126.13	7.22	326.93	0.00	0.41	-0.10	0.42	2.11	0.00
117.6	5.00	331.11	0.00	0.00	5.00	-28.89	5.00	-28.89	5.00	-28.89	5.00	-28.89	-125.70	6.97	329.06	0.00	0.41	-0.10	0.42	0.56	0.00
117.7	5.00	331.61	0.00	0.00	5.00	-28.39	5.00	-28.39	5.00	-28.39	5.00	-28.39	-125.26	6.73	331.11	0.00	0.41	-0.11	0.42	-0.92	0.00
117.8	5.00	330.63	0.00	0.00	5.00	-29.37	5.00	-29.37	5.00	-29.37	5.00	-29.37	-124.82	6.49	331.61	0.00	0.40	-0.12	0.42	-0.84	0.00
117.9	5.00	329.73	0.00	0.00	5.00	-30.27	5.00	-30.27	5.00	-30.27	5.00	-30.27	-124.39	6.24	330.63	0.00	0.40	-0.12	0.42	0.71	0.00
118	5.00	330.38	0.00	0.00	5.00	-29.62	5.00	-29.62	5.00	-29.62	5.00	-29.62	-123.95	5.99	329.73	0.00	0.41	-0.10	0.42	2.18	0.00
118.1	5.00	332.50	0.00	0.00	5.00	-27.50	5.00	-27.50	5.00	-27.50	5.00	-27.50	-123.52	5.74	330.38	0.00	0.42	-0.09	0.43	2.10	0.00
118.2	5.00	334.54	0.00	0.00	5.00	-25.46	5.00	-25.46	5.00	-25.46	5.00	-25.46	-123.08	5.51	332.50	0.00	0.42	-0.09	0.43	0.56	0.00
118.3	5.00	335.03	0.00	0.00	5.00	-24.97	5.00	-24.97	5.00	-24.97	5.00	-24.97	-122.62	5.30	334.54	0.00	0.42	-0.09	0.43	-0.91	0.00
118.4	5.00	334.05	0.00	0.00	5.00	-25.95	5.00	-25.95	5.00	-25.95	5.00	-25.95	-122.17	5.09	335.03	0.00	0.41	-0.10	0.43	-0.82	0.00
118.5	5.00	333.17	0.00	0.00	5.00	-26.83	5.00	-26.83	5.00	-26.83	5.00	-26.83	-121.72	4.87	334.05	0.00	0.41	-0.10	0.43	0.72	0.00
118.6	5.00	333.83	0.00	0.00	5.00	-26.17	5.00	-26.17	5.00	-26.17	5.00	-26.17	-121.28	4.64	333.17	0.00	0.42	-0.09	0.43	2.18	0.00
118.7	5.00	335.94	0.00	0.00	5.00	-24.06	5.00	-24.06	5.00	-24.06	5.00	-24.06	-120.83	4.42	333.83	0.00	0.43	-0.07	0.43	2.09	0.00

118.8	5.00	337.97	0.00	0.00	5.00	-22.03	5.00	-22.03	5.00	-22.03	5.00	-22.03	-120.37	4.22	335.94	0.00	0.43	-0.07	0.44	0.55	0.00
118.9	5.00	338.45	0.00	0.00	5.00	-21.55	5.00	-21.55	5.00	-21.55	5.00	-21.55	-119.91	4.03	337.97	0.00	0.43	-0.08	0.43	-0.90	0.00
119	5.00	337.48	0.00	0.00	5.00	-22.52	5.00	-22.52	5.00	-22.52	5.00	-22.52	-119.44	3.85	338.45	0.00	0.42	-0.09	0.43	-0.80	0.00
119.1	5.00	336.61	0.00	0.00	5.00	-23.39	5.00	-23.39	5.00	-23.39	5.00	-23.39	-118.98	3.65	337.48	0.00	0.42	-0.08	0.43	0.73	0.00
119.2	5.00	337.28	0.00	0.00	5.00	-22.72	5.00	-22.72	5.00	-22.72	5.00	-22.72	-118.52	3.46	336.61	0.00	0.43	-0.07	0.44	2.18	0.00
119.3	5.00	339.38	0.00	0.00	5.00	-20.62	5.00	-20.62	5.00	-20.62	5.00	-20.62	-118.06	3.26	337.28	0.00	0.44	-0.05	0.44	2.09	0.00
119.4	5.00	341.39	0.00	0.00	5.00	-18.61	5.00	-18.61	5.00	-18.61	5.00	-18.61	-117.59	3.09	339.38	0.00	0.44	-0.05	0.44	0.55	0.00
119.5	5.00	341.87	0.00	0.00	5.00	-18.13	5.00	-18.13	5.00	-18.13	5.00	-18.13	-117.12	2.93	341.39	0.00	0.44	-0.06	0.44	-0.88	0.00
119.6	5.00	340.91	0.00	0.00	5.00	-19.09	5.00	-19.09	5.00	-19.09	5.00	-19.09	-116.64	2.77	341.87	0.00	0.43	-0.07	0.44	-0.78	0.00
119.7	5.00	340.05	0.00	0.00	5.00	-19.95	5.00	-19.95	5.00	-19.95	5.00	-19.95	-116.17	2.61	340.91	0.00	0.43	-0.07	0.44	0.75	0.00
119.8	5.00	340.73	0.00	0.00	5.00	-19.27	5.00	-19.27	5.00	-19.27	5.00	-19.27	-115.70	2.44	340.05	0.00	0.44	-0.05	0.44	2.18	0.00
119.9	5.00	342.82	0.00	0.00	5.00	-17.18	5.00	-17.18	5.00	-17.18	5.00	-17.18	-115.23	2.27	340.73	0.00	0.44	-0.04	0.45	2.08	0.00
120	5.00	344.82	0.00	0.00	5.00	-15.18	5.00	-15.18	5.00	-15.18	5.00	-15.18	-114.75	2.12	342.82	0.00	0.45	-0.03	0.45	0.55	0.00
120.1	5.00	345.29	0.00	0.00	5.00	-14.71	5.00	-14.71	5.00	-14.71	5.00	-14.71	-114.27	1.99	344.82	0.00	0.44	-0.04	0.45	-0.87	0.00
120.2	5.00	344.34	0.00	0.00	5.00	-15.66	5.00	-15.66	5.00	-15.66	5.00	-15.66	-113.78	1.87	345.29	0.00	0.44	-0.05	0.44	-0.77	0.00
120.3	5.00	343.49	0.00	0.00	5.00	-16.51	5.00	-16.51	5.00	-16.51	5.00	-16.51	-113.30	1.73	344.34	0.00	0.44	-0.05	0.45	0.76	0.00
120.4	5.00	344.17	0.00	0.00	5.00	-15.83	5.00	-15.83	5.00	-15.83	5.00	-15.83	-112.82	1.59	343.49	0.00	0.45	-0.03	0.45	2.17	0.00
120.5	5.00	346.26	0.00	0.00	5.00	-13.74	5.00	-13.74	5.00	-13.74	5.00	-13.74	-112.34	1.45	344.17	0.00	0.45	-0.02	0.45	2.07	0.00
120.6	5.00	348.25	0.00	0.00	5.00	-11.75	5.00	-11.75	5.00	-11.75	5.00	-11.75	-111.86	1.33	346.26	0.00	0.45	-0.02	0.45	0.55	0.00
120.7	5.00	348.71	0.00	0.00	5.00	-11.29	5.00	-11.29	5.00	-11.29	5.00	-11.29	-111.37	1.23	348.25	0.00	0.45	-0.03	0.45	-0.86	0.00
120.8	5.00	347.77	0.00	0.00	5.00	-12.23	5.00	-12.23	5.00	-12.23	5.00	-12.23	-110.88	1.14	348.71	0.00	0.45	-0.04	0.45	-0.75	0.00
120.9	5.00	346.94	0.00	0.00	5.00	-13.06	5.00	-13.06	5.00	-13.06	5.00	-13.06	-110.39	1.03	347.77	0.00	0.45	-0.03	0.45	0.77	0.00
121	5.00	347.62	0.00	0.00	5.00	-12.38	5.00	-12.38	5.00	-12.38	5.00	-12.38	-109.90	0.92	346.94	0.00	0.45	-0.02	0.45	2.17	0.00
121.1	5.00	349.70	0.00	0.00	5.00	-10.30	5.00	-10.30	5.00	-10.30	5.00	-10.30	-109.41	0.81	347.62	0.00	0.46	0.00	0.46	2.06	0.00
121.2	5.00	351.68	0.00	0.00	5.00	-8.32	5.00	-8.32	5.00	-8.32	5.00	-8.32	-108.92	0.72	349.70	0.00	0.46	0.00	0.46	0.55	0.00
121.3	5.00	352.13	0.00	0.00	5.00	-7.87	5.00	-7.87	5.00	-7.87	5.00	-7.87	-108.43	0.65	351.68	0.00	0.46	-0.01	0.46	-0.85	0.00
121.4	5.00	351.20	0.00	0.00	5.00	-8.80	5.00	-8.80	5.00	-8.80	5.00	-8.80	-107.93	0.58	352.13	0.00	0.46	-0.02	0.46	-0.73	0.00
121.5	5.00	350.38	0.00	0.00	5.00	-9.62	5.00	-9.62	5.00	-9.62	5.00	-9.62	-107.44	0.50	351.20	0.00	0.46	-0.01	0.46	0.78	0.00
121.6	5.00	351.07	0.00	0.00	5.00	-8.93	5.00	-8.93	5.00	-8.93	5.00	-8.93	-106.94	0.42	350.38	0.00	0.46	0.00	0.46	2.17	0.00
121.7	5.00	353.14	0.00	0.00	5.00	-6.86	5.00	-6.86	5.00	-6.86	5.00	-6.86	-106.45	0.34	351.07	0.00	0.46	0.02	0.46	2.06	0.00
121.8	5.00	355.10	0.00	0.00	5.00	-4.90	5.00	-4.90	5.00	-4.90	5.00	-4.90	-105.95	0.28	353.14	0.00	0.46	0.02	0.46	0.55	0.00
121.9	5.00	355.56	0.00	0.00	5.00	-4.44	5.00	-4.44	5.00	-4.44	5.00	-4.44	-105.46	0.24	355.10	0.00	0.46	0.01	0.46	-0.84	0.00
122	5.00	354.63	0.00	0.00	5.00	-5.37	5.00	-5.37	5.00	-5.37	5.00	-5.37	-104.96	0.20	355.56	0.00	0.46	0.00	0.46	-0.72	0.00
122.1	5.00	353.82	0.00	0.00	5.00	-6.18	5.00	-6.18	5.00	-6.18	5.00	-6.18	-104.46	0.15	354.63	0.00	0.46	0.01	0.46	0.79	0.00
122.2	5.00	354.51	0.00	0.00	5.00	-5.49	5.00	-5.49	5.00	-5.49	5.00	-5.49	-103.96	0.10	353.82	0.00	0.46	0.02	0.46	2.17	0.00
122.3	5.00	356.58	0.00	0.00	5.00	-3.42	5.00	-3.42	5.00	-3.42	5.00	-3.42	-103.46	0.05	354.51	0.00	0.46	0.04	0.47	2.05	0.00
122.4	5.00	358.53	0.00	0.00	5.00	-1.47	5.00	-1.47	5.00	-1.47	5.00	-1.47	-102.96	0.02	356.58	0.00	0.46	0.04	0.47	0.55	0.00
122.5	5.00	358.98	0.00	0.00	5.00	-1.02	5.00	-1.02	5.00	-1.02	5.00	-1.02	-102.47	0.01	358.53	0.00	0.46	0.03	0.47	-0.83	0.00
122.6	5.00	358.06	0.00	0.00	5.00	-1.94	5.00	-1.94	5.00	-1.94	5.00	-1.94	-101.97	0.00	358.98	0.00	0.46	0.02	0.47	-0.70	0.00
122.7	5.00	357.26	0.00	0.00	5.00	-2.74	5.00	-2.74	5.00	-2.74	5.00	-2.74	-101.47	-0.02	358.06	0.00	0.47	0.03	0.47	0.80	0.00
122.8	5.00	357.96	0.00	0.00	5.00	-2.04	5.00	-2.04	5.00	-2.04	5.00	-2.04	-100.97	-0.04	357.26	0.00	0.47	0.04	0.47	2.17	0.00
122.9	5.00	360.02	0.00	0.00	5.00	0.02	5.00	0.02	5.00	0.02	5.00	0.02	-100.47	-0.06	357.96	0.00	0.47	0.06	0.47	2.04	0.00
123	5.00	361.96	0.00	0.00	5.00	1.96	5.00	1.96	5.00	1.96	5.00	1.96	-99.97	-0.06	360.02	0.00	0.47	0.06	0.47	-0.02	0.00
123.1	5.00	361.83	0.00	0.00	5.00	1.83	5.00	1.83	5.00	1.83	5.00	1.83	-99.47	-0.04	361.96	0.00	0.47	0.04	0.47	-1.96	0.00
123.2	5.00	359.77	0.00	0.00	5.00	-0.23	5.00	-0.23	5.00	-0.23	5.00	-0.23	-98.97	-0.03	361.83	0.00	0.47	0.03	0.47	-1.83	0.00
123.3	5.00	357.84	0.00	0.00	5.00	-2.16	5.00	-2.16	5.00	-2.16	5.00	-2.16	-98.47	-0.03	359.77	0.00	0.47	0.03	0.47	0.23	0.00
123.4	5.00	357.97	0.00	0.00	5.00	-2.03	5.00	-2.03	5.00	-2.03	5.00	-2.03	-97.97	-0.05	357.84	0.00	0.47	0.05	0.47	2.16	0.00
123.5	5.00	360.03	0.00	0.00	5.00	0.03	5.00	0.03	5.00	0.03	5.00	0.03	-97.47	-0.06	357.97	0.00	0.47	0.06	0.47	2.03	0.00
123.6	5.00	361.95	0.00	0.00	5.00	1.95	5.00	1.95	5.00	1.95	5.00	1.95	-96.97	-0.06	360.03	0.00	0.47	0.06	0.47	-0.03	0.00
123.7	5.00	361.82	0.00	0.00	5.00	1.82	5.00	1.82	5.00	1.82	5.00	1.82	-96.47	-0.05	361.95	0.00	0.47	0.05	0.47	-1.95	0.00
123.8	5.00	359.76	0.00	0.00	5.00	-0.24	5.00	-0.24	5.00	-0.24	5.00	-0.24	-95.97	-0.03	361.82	0.00	0.47	0.03	0.47	-1.82	0.00
123.9	5.00	357.84	0.00	0.00	5.00	-2.16	5.00	-2.16	5.00	-2.16	5.00	-2.16	-95.47	-0.03	359.76	0.00	0.47	0.03	0.47	0.24	0.00
124	5.00	357.98	0.00	0.00	5.00	-2.02	5.00	-2.02	5.00	-2.02	5.00	-2.02	-94.97	-0.05	357.84	0.00	0.47	0.05	0.47	2.16	0.00
124.1	5.00	360.03	0.00	0.00	5.00	0.03	5.00	0.03	5.00	0.03	5.00	0.03	-94.47	-0.07	357.98	0.00	0.47	0.07	0.47	2.02	0.00
124.2	5.00	361.94	0.00	0.00	5.00	1.94	5.00	1.94	5.00	1.94	5.00	1.94	-93.97	-0.07	360.03	0.00	0.47	0.07	0.47	-0.03	0.00
124.3	5.00	361.80	0.00	0.00	5.00	1.80	5.00	1.80	5.00	1.80	5.00	1.80	-93.47	-0.05	361.94	0.00	0.47	0.05	0.47	-1.94	0.00
124.4	5.00	359.75	0.00	0.00	5.00	-0.25	5.00	-0.25	5.00	-0.25	5.00	-0.25	-92.97	-0.04	361.80	0.00	0.47	0.04	0.47	-1.80	0.00
124.5	5.00	357.85	0.00	0.00	5.00	-2.15	5.00	-2.15	5.00	-2.15	5.00	-2.15	-92.47	-0.04	359.75	0.00	0.47	0.04	0.47	0.25	0.00
124.6	5.00	357.99	0.00	0.00	5.00	-2.01	5.00	-2.01	5.00	-2.01	5.00	-2.01	-91.97	-0.06	357.85	0.00	0.47	0.06	0.47	2.15	0.00
124.7	5.00	360.04	0.00	0.00	5.00	0.04	5.00	0.04	5.00	0.04	5.00	0.04	-91.47	-0.07	357.99	0.00	0.47	0.07	0.48	2.01	0.00
124.8	5.00	361.93	0.00	0.00	5.00	1.93	5.00	1.93	5.00	1.93	5.00	1.93	-90.97	-0.07	360.04	0.00	0.47	0.07	0.48	-0.04	0.00
124.9	5.00	361.79	0.00	0.00	5.00	1.79	5.00	1.79	5.00	1.79	5.00	1.79	-90.47	-0.06	361.93	0.00	0.47	0.06	0.47	-1.93	0.00
125	5.00	359.75	0.00	0.00	5.00	-0.25	5.00	-0.25	5.00	-0.25	5.00	-0.25	-89.97	-0.04	361.79	0.00	0.47	0.04	0.47	-1.79	0.00
125.1	5.00	357.86	0.00	0.00	5.00	-2.14	5.00	-2.14	5.00	-2.14	5.00	-2.14	-89.47	-0.04	359.75	0.00	0.47	0.04	0.47	0.25	0.00
125.2	5.00	358.01	0.00	0.00	5.00	-1.99	5.00	-1.99	5.00	-1.99	5.00	-1.99	-88.97	-0.06	357.86	0.00	0.47	0.06	0.48	2.14	0.00
125.3	5.00	360.04	0.00	0.00	5.00	0.04	5.00	0.04	5.00	0.04	5.00	0.04	-88.47	-0.08	358.01	0.00	0.47	0.08	0.48	1.99	0.00
125.4	5.00	361.93	0.00	0.00	5.00	1.93	5.00	1.93	5.00	1.93	5.00	1.93	-87.97	-0.08	360.04	0.00	0.47	0.08	0.48	-0.04	0.00
125.5	5.00	361.77	0.00	0.00	5.00	1.77	5.00	1.77	5.00	1.77	5.00	1.77	-87.47	-0.06	361.93	0.00	0.47	0.06	0.48	-1.93	0.00
125.6	5.00	359.74	0.00	0.00	5.00	-0.26	5.00	-0.26	5.00	-0.26	5.00	-0.26	-86.97	-0.05	361.77	0.00	0.47	0.05	0.47	-1.77	0.00
125.7	5.00	357.86	0.00	0.00	5.00	-2.14	5.00	-2.14	5.00	-2.14	5.00	-2.14	-86.47	-0.05	359.74	0.00	0.47	0.05	0.47	0.26	0.00
125.8	5.00	358.02	0.00	0.00	5.00	-1.98	5.00	-1.98	5.00	-1.98	5.00	-1.98	-85.97	-0.07	357.86	0.00	0.47	0.07	0.48	2.14	0.00
125.9	5.00	360.05	0.00	0.00	5.00	0.05	5.00	0.05	5.00	0.05	5.00	0.05	-85.47	-0.09	358.02	0.00	0.47	0.09	0.48	1.98	0.00
126	5.00	361.92	0.00	0.00	5.00	1.92	5.00	1.92	5.00	1.92	5.00	1.92	-84.97	-0.09	360.05	0.00	0.47	0.09	0.48	-0.05	0.00
126.1	5.00	361.76	0.00	0.00	5.00	1.76	5.00	1.76	5.00	1.76	5.00	1.76	-84.47	-0.07	361.92	0.00	0.47	0.07	0.48	-1.92	0.00
126.2	5.00	359.73	0.00	0.00	5.00	-0.27	5.00	-0.27	5.00	-0.27	5.00	-0.27	-83.97	-0.05	361.76	0.00	0.47	0.05	0.48	-1.76	0.00
126.3	5.00	357.87	0.00	0.00	5.00	-2.13	5.00	-2.13	5.00	-2.13	5.00	-2.13	-83.47	-0.06	359.73	0.00	0.47	0.06	0.48	0.27	0.00
126.4	5.00	358.03	0.00	0.00	5.00	-1.97	5.00	-1.97	5.00	-1.97	5.00	-1.97	-82.97	-0.07	357.87	0.00	0.47	0.07	0.48	2.13	0.00
126.5	5.00	360.05	0.00	0.00	5.00	0.05	5.00	0.05	5.00	0.05	5.00	0.05	-82.47	-0.09	358.03	0.00	0.47	0.09	0.48	1.97	0.00
126.6	5.00	361.91	0.00	0.00	5.00	1.91	5.00	1.91	5.00	1.91	5.00	1.91	-81.97	-0.09	360.05	0.00	0.47	0.09	0.48	-0.05	0.00
126.7	5.00	361.74	0.00	0.00	5.00	1.74	5.00	1.74	5.00	1.74	5.00	1.74	-81.47	-0.07	361.91	0.00	0.47	0.07	0.48	-1.91	0.00
126.8	5.00	359.73	0.00	0.00	5.00	-0.27	5.00	-0.27	5.00	-0.27	5.00	-0.27	-80.97	-0.06	361.74	0.00	0.47	0.06	0.48	-1.74	0.00
126.9	5.00	357.88	0.00	0.00	5.00	-2.12	5.00	-2.12	5.00	-2.12	5.00	-2.12	-80.47	-0.06	359.73	0.00	0.47	0.06	0.48	0.27	0.00
127	5.00	358.04	0.00	0.00	5.00	-1.96	5.00	-1.96	5.00	-1.96	5.00	-1.96	-79.97	-0.08	357.88	0.00	0.47	0.08	0.48	2.12	0.00
127.1	5.00	360.06	0.00	0.00	5.00	0.06	5.00	0.06	5.00	0.06	5.00	0.06	-79.47	-0.10	358.04	0.00	0.47	0.10	0.48	1.96	0.00
127.2	5.00	361.90	0.00	0.00	5.00	1.90	5.00	1.90	5.00	1.90	5.00	1.90	-78.97	-0.10	360.06	0.00	0.47	0.10	0.48	-0.06	0.00
127.3	5.00	361.73	0.00	0.00	5.00	1.73	5.00	1.73	5.00	1.73	5.00	1.73	-78.47	-0.08	361.90	0.00	0.47	0.08	0.48	-1.90	0.00
127.4	5.00	359.72	0.00	0.00	5.00	-0.28	5.00	-0.28	5.00	-0.28	5.00	-0.28	-77.98	-0.06	361.73	0.00	0.48	0.06	0.48	-1.73	0.00
127.5	5.00	357.88	0.00	0.00	5.00	-2.12	5.00	-2.12	5.00	-2.12	5.00	-2.12	-77.48	-0.07	359.72	0.00	0.48	0.07	0.48	0.28	0.00
127.6	5.00	358.06	0.00	0.00	5.00	-1.94	5.00	-1.94	5.00	-1.94	5.00	-1.94	-76.98	-0.09	357.88	0.00	0.48	0.09	0.48	2.12	0.00
127.7	5.00	360.06	0.00	0.00	5.00	0.06	5.00	0.06	5.00	0.06	5.00	0.06	-76.48	-0.10	358.06	0.00	0.48	0.10	0.49	1.94	0.00
127.8	5.00	361.89	0.00	0.00	5.00	1.89	5.00	1.89	5.00	1.89	5.00	1.89	-75.98	-0.10	360.06	0.00	0.48	0.10	0.49	-0.06	0.00
127.9	5.00	361.71	0.00	0.00	5.00	1.71	5.00	1.71	5.00	1.71	5.00	1.71	-75.48	-0.09	361.89	0.00	0.48	0.09	0.48	-1.89	0.00
128	5.00	359.71	0.00	0.00	5.00	-0.29	5.00	-0.29	5.00	-0.29	5.00	-0.29	-74.98	-0.07	361.71	0.00	0.48	0.07	0.48	-1.71	0.00
128.1	5.00	357.89	0.00	0.00	5.00	-2.11	5.00	-2.11	5.00	-2.11	5.00	-2.11	-74.48	-0.07	359.71	0.00	0.48	0.07	0.48	0.29	0.00
128.2	5.00	358.07	0.00	0.00	5.00	-1.93	5.00	-1.93	5.00	-1.93	5.00	-1.93	-73.98	-0.09	357.89	0.00	0.48	0.09	0.49	2.11	0.00
128.3	5.00	360.07	0.00	0.00	5.00	0.07	5.00	0.07	5.00	0.07	5.00	0.07	-73.48	-0.11	358.07	0.00	0.48	0.11	0.49	1.93	0.00
128.4	5.00	361.88	0.00	0.00	5.00	1.88	5.00	1.88	5.00	1.88	5.00	1.88	-72.98	-0.11	360.07	0.00	0.48	0.11	0.49	-0.07	0.00
128.5	5.00	361.70	0.00	0.00	5.00	1.70	5.00	1.70	5.00	1.70	5.00	1.70	-72.48	-0.09	361.88	0.00	0.48	0.09	0.49	-1.88	0.00
128.6	5.00	359.70	0.00	0.00	5.00	-0.30	5.00	-0.30	5.00	-0.30	5.00	-0.30	-71.98	-0.08	361.70	0.00	0.48	0.08	0.48	-1.70	0.00
128.7	5.00	357.89	0.00	0.00	5.00	-2.11	5.00	-2.11	5.00	-2.11	5.00	-2.11	-71.48	-0.08	359.70	0.00	0.48	0.08	0.48	0.30	0.00
128.8	5.00	358.08	0.00	0.00	5.00	-1.92	5.00	-1.92	5.00	-1.92	5.00	-1.92	-70.98	-0.10	357.89	0.00	0.48	0.10	0.49	2.11	0.00
128.9	5.00	360.07	0.00	0.00	5.00	0.07	5.00	0.07	5.00	0.07	5.00	0.07	-70.48	-0.11	358.08	0.00	0.48	0.11	0.49	1.92	0.00
129	5.00	361.87	0.00	0.00	5.00	1.87	5.00	1.87	5.00	1.87	5.00	1.87	-69.98	-0.11	360.07	0.00	0.48	0.11	0.49	-0.07	0.00
129.1	5.00	361.68	0.00	0.00	5.00	1.68	5.00	1.68	5.00	1.68	5.00	1.68	-69.48	-0.10	361.87	0.00	0.48	0.10	0.49	-1.87	0.00
129.2	5.00	359.70	0.00	0.00	5.00	-0.30	5.00	-0.30	5.00	-0.30	5.00	-0.30	-68.98	-0.08	361.68	0.00	0.48	0.08	0.49	-1.68	0.00
129.3	5.00	357.90	0.00	0.00	5.00	-2.10	5.00	-2.10	5.00	-2.10	5.00	-2.10	-68.48	-0.09	359.70	0.00	0.48	0.09	0.49	0.30	0.00
129.4	5.00	358.09	0.00	0.00	5.00	-1.91	5.00	-1.91	5.00	-1.91	5.00	-1.91	-67.98	-0.10	357.90	0.00	0.48	0.10	0.49	2.10	0.00
129.5	5.00	360.07	0.00	0.00	5.00	0.07	5.00	0.07	5.00	0.07	5.00	0.07	-67.48	-0.12	358.09	0.00	0.48	0.12	0.49	1.91	0.00
129.6	5.00	361.86	0.00	0.00	5.00	1.86	5.00	1.86	5.00	1.86	5.00	1.86	-66.98	-0.12	360.07	0.00	0.48	0.12	0.49	-0.07	0.00
129.7	5.00	361.67	0.00	0.00	5.00	1.67	5.00	1.67	5.00	1.67	5.00	1.67	-66.48	-0.10	361.86	0.00	0.48	0.10	0.49	-1.86	0.00
129.8	5.00	359.69	0.00	0.00	5.00	-0.31	5.00	-0.31	5.00	-0.31	5.00	-0.31	-65.98	-0.09	361.67	0.00	0.48	0.09	0.49	-1.67	0.00
129.9	5.00	357.91	0.00	0.00	5.00	-2.09	5.00	-2.09	5.00	-2.09	5.00	-2.09	-65.48	-0.09	359.69	0.00	0.48	0.09	0.49	0.31	0.00
130	5.00	358.10	0.00	0.00	5.00	-1.90	5.00	-1.90	5.00	-1.90	5.00	-1.90	-64.98	-0.11	357.91	0.00	0.48	0.11	0.49	2.09	0.00
130.1	5.00	360.08	0.00	0.00	5.00	0.08	5.00	0.08	5.00	0.08	5.00	0.08	-64.48	-0.13	358.10	0.00	0.48	0.13	0.50	1.90	0.00
130.2	5.00	361.85	0.00	0.00	5.00	1.85	5.00	1.85	5.00	1.85	5.00	1.85	-63.98	-0.13	360.08	0.00	0.48	0.13	0.50	-0.08	0.00
130.3	5.00	361.65	0.00	0.00	5.00	1.65	5.00	1.65	5.00	1.65	5.00	1.65	-63.48	-0.11	361.85	0.00	0.48	0.11	0.49	-1.85	0.00
130.4	5.00	359.68	0.00	0.00	5.00	-0.32	5.00	-0.32	5.00	-0.32	5.00	-0.32	-62.98	-0.10	361.65	0.00	0.48	0.10	0.49	-1.65	0.00
130.5	5.00	357.91	0.00	0.00	5.00	-2.09	5.00	-2.09	5.00	-2.09	5.00	-2.09	-62.48	-0.10	359.68	0.00	0.48	0.10	0.49	0.32	0.00
130.6	5.00	358.11	0.00	0.00	5.00	-1.89	5.00	-1.89	5.00	-1.89	5.00	-1.89	-61.98	-0.12	357.91	0.00	0.48	0.12	0.49	2.09	0.00

130.7	5.00	360.08	0.00	0.00	5.00	0.08	5.00	0.08	5.00	0.08	5.00	0.08	-61.48	-0.13	358.11	0.00	0.48	0.13	0.50	1.89	0.00
130.8	5.00	361.84	0.00	0.00	5.00	1.84	5.00	1.84	5.00	1.84	5.00	1.84	-60.98	-0.13	360.08	0.00	0.48	0.13	0.50	-0.08	0.00
130.9	5.00	361.64	0.00	0.00	5.00	1.64	5.00	1.64	5.00	1.64	5.00	1.64	-60.48	-0.12	361.84	0.00	0.48	0.12	0.50	-1.84	0.00
131	5.00	359.68	0.00	0.00	5.00	-0.32	5.00	-0.32	5.00	-0.32	5.00	-0.32	-59.98	-0.10	361.64	0.00	0.48	0.10	0.49	-1.64	0.00
131.1	5.00	357.92	0.00	0.00	5.00	-2.08	5.00	-2.08	5.00	-2.08	5.00	-2.08	-59.48	-0.10	359.68	0.00	0.48	0.10	0.49	0.32	0.00
131.2	5.00	358.13	0.00	0.00	5.00	-1.87	5.00	-1.87	5.00	-1.87	5.00	-1.87	-58.98	-0.12	357.92	0.00	0.48	0.12	0.50	2.08	0.00
131.3	5.00	360.09	0.00	0.00	5.00	0.09	5.00	0.09	5.00	0.09	5.00	0.09	-58.48	-0.14	358.13	0.00	0.48	0.14	0.50	1.87	0.00
131.4	5.00	361.84	0.00	0.00	5.00	1.84	5.00	1.84	5.00	1.84	5.00	1.84	-57.98	-0.14	360.09	0.00	0.48	0.14	0.50	-0.09	0.00
131.5	5.00	361.63	0.00	0.00	5.00	1.63	5.00	1.63	5.00	1.63	5.00	1.63	-57.48	-0.12	361.84	0.00	0.48	0.12	0.50	-1.84	0.00
131.6	5.00	359.67	0.00	0.00	5.00	-0.33	5.00	-0.33	5.00	-0.33	5.00	-0.33	-56.98	-0.11	361.63	0.00	0.48	0.11	0.49	-1.63	0.00
131.7	5.00	357.93	0.00	0.00	5.00	-2.07	5.00	-2.07	5.00	-2.07	5.00	-2.07	-56.48	-0.11	359.67	0.00	0.48	0.11	0.50	0.33	0.00
131.8	5.00	358.14	0.00	0.00	5.00	-1.86	5.00	-1.86	5.00	-1.86	5.00	-1.86	-55.98	-0.13	357.93	0.00	0.48	0.13	0.50	2.07	0.00
131.9	5.00	360.09	0.00	0.00	5.00	0.09	5.00	0.09	5.00	0.09	5.00	0.09	-55.48	-0.15	358.14	0.00	0.48	0.15	0.50	1.86	0.00
132	5.00	361.83	0.00	0.00	5.00	1.83	5.00	1.83	5.00	1.83	5.00	1.83	-54.98	-0.14	360.09	0.00	0.48	0.14	0.50	-0.09	0.00
132.1	5.00	361.61	0.00	0.00	5.00	1.61	5.00	1.61	5.00	1.61	5.00	1.61	-54.48	-0.13	361.83	0.00	0.48	0.13	0.50	-1.83	0.00
132.2	5.00	359.66	0.00	0.00	5.00	-0.34	5.00	-0.34	5.00	-0.34	5.00	-0.34	-53.98	-0.11	361.61	0.00	0.48	0.11	0.50	-1.61	0.00
132.3	5.00	357.93	0.00	0.00	5.00	-2.07	5.00	-2.07	5.00	-2.07	5.00	-2.07	-53.48	-0.12	359.66	0.00	0.48	0.12	0.50	0.34	0.00
132.4	5.00	358.15	0.00	0.00	5.00	-1.85	5.00	-1.85	5.00	-1.85	5.00	-1.85	-52.98	-0.14	357.93	0.00	0.48	0.14	0.50	2.07	0.00
132.5	5.00	360.09	0.00	0.00	5.00	0.09	5.00	0.09	5.00	0.09	5.00	0.09	-52.48	-0.15	358.15	0.00	0.48	0.15	0.51	1.85	0.00
132.6	5.00	361.82	0.00	0.00	5.00	1.82	5.00	1.82	5.00	1.82	5.00	1.82	-51.98	-0.15	360.09	0.00	0.48	0.15	0.51	-0.09	0.00
132.7	5.00	361.60	0.00	0.00	5.00	1.60	5.00	1.60	5.00	1.60	5.00	1.60	-51.48	-0.13	361.82	0.00	0.48	0.13	0.50	-1.82	0.00
132.8	5.00	359.66	0.00	0.00	5.00	-0.34	5.00	-0.34	5.00	-0.34	5.00	-0.34	-50.98	-0.12	361.60	0.00	0.48	0.12	0.50	-1.60	0.00
132.9	5.00	357.94	0.00	0.00	5.00	-2.06	5.00	-2.06	5.00	-2.06	5.00	-2.06	-50.48	-0.12	359.66	0.00	0.48	0.12	0.50	0.34	0.00
133	5.00	358.16	0.00	0.00	5.00	-1.84	5.00	-1.84	5.00	-1.84	5.00	-1.84	-49.99	-0.14	357.94	0.00	0.49	0.14	0.51	2.06	0.00
133.1	5.00	360.10	0.00	0.00	5.00	0.10	5.00	0.10	5.00	0.10	5.00	0.10	-49.49	-0.16	358.16	0.00	0.49	0.16	0.51	1.84	0.00
133.2	5.00	361.81	0.00	0.00	5.00	1.81	5.00	1.81	5.00	1.81	5.00	1.81	-48.99	-0.16	360.10	0.00	0.49	0.16	0.51	-0.10	0.00
133.3	5.00	361.58	0.00	0.00	5.00	1.58	5.00	1.58	5.00	1.58	5.00	1.58	-48.49	-0.14	361.81	0.00	0.49	0.14	0.51	-1.81	0.00
133.4	5.00	359.65	0.00	0.00	5.00	-0.35	5.00	-0.35	5.00	-0.35	5.00	-0.35	-47.99	-0.13	361.58	0.00	0.49	0.13	0.50	-1.58	0.00
133.5	5.00	357.94	0.00	0.00	5.00	-2.06	5.00	-2.06	5.00	-2.06	5.00	-2.06	-47.49	-0.13	359.65	0.00	0.49	0.13	0.50	0.35	0.00
133.6	5.00	358.17	0.00	0.00	5.00	-1.83	5.00	-1.83	5.00	-1.83	5.00	-1.83	-46.99	-0.15	357.94	0.00	0.49	0.15	0.51	2.06	0.00
133.7	5.00	360.10	0.00	0.00	5.00	0.10	5.00	0.10	5.00	0.10	5.00	0.10	-46.49	-0.16	358.17	0.00	0.49	0.16	0.51	1.83	0.00
133.8	5.00	361.80	0.00	0.00	5.00	1.80	5.00	1.80	5.00	1.80	5.00	1.80	-45.99	-0.16	360.10	0.00	0.49	0.16	0.51	-0.10	0.00
133.9	5.00	361.57	0.00	0.00	5.00	1.57	5.00	1.57	5.00	1.57	5.00	1.57	-45.49	-0.15	361.80	0.00	0.49	0.15	0.51	-1.80	0.00
134	5.00	359.64	0.00	0.00	5.00	-0.36	5.00	-0.36	5.00	-0.36	5.00	-0.36	-44.99	-0.13	361.57	0.00	0.49	0.13	0.51	-1.57	0.00
134.1	5.00	357.95	0.00	0.00	5.00	-2.05	5.00	-2.05	5.00	-2.05	5.00	-2.05	-44.49	-0.14	359.64	0.00	0.49	0.14	0.51	0.36	0.00
134.2	5.00	358.18	0.00	0.00	5.00	-1.82	5.00	-1.82	5.00	-1.82	5.00	-1.82	-43.99	-0.16	357.95	0.00	0.49	0.16	0.51	2.05	0.00
134.3	5.00	360.10	0.00	0.00	5.00	0.10	5.00	0.10	5.00	0.10	5.00	0.10	-43.49	-0.17	358.18	0.00	0.49	0.17	0.52	1.82	0.00
134.4	5.00	361.79	0.00	0.00	5.00	1.79	5.00	1.79	5.00	1.79	5.00	1.79	-42.99	-0.17	360.10	0.00	0.49	0.17	0.52	-0.10	0.00
134.5	5.00	361.55	0.00	0.00	5.00	1.55	5.00	1.55	5.00	1.55	5.00	1.55	-42.49	-0.15	361.79	0.00	0.49	0.15	0.51	-1.79	0.00
134.6	5.00	359.64	0.00	0.00	5.00	-0.36	5.00	-0.36	5.00	-0.36	5.00	-0.36	-41.99	-0.14	361.55	0.00	0.49	0.14	0.51	-1.55	0.00
134.7	5.00	357.96	0.00	0.00	5.00	-2.04	5.00	-2.04	5.00	-2.04	5.00	-2.04	-41.49	-0.14	359.64	0.00	0.49	0.14	0.51	0.36	0.00
134.8	5.00	358.19	0.00	0.00	5.00	-1.81	5.00	-1.81	5.00	-1.81	5.00	-1.81	-40.99	-0.16	357.96	0.00	0.49	0.16	0.51	2.04	0.00
134.9	5.00	360.11	0.00	0.00	5.00	0.11	5.00	0.11	5.00	0.11	5.00	0.11	-40.49	-0.18	358.19	0.00	0.49	0.18	0.52	1.81	0.00
135	5.00	361.78	0.00	0.00	5.00	1.78	5.00	1.78	5.00	1.78	5.00	1.78	-39.99	-0.18	360.11	0.00	0.49	0.18	0.52	-0.11	0.00
135.1	5.00	361.54	0.00	0.00	5.00	1.54	5.00	1.54	5.00	1.54	5.00	1.54	-39.49	-0.16	361.78	0.00	0.49	0.16	0.51	-1.78	0.00
135.2	5.00	359.63	0.00	0.00	5.00	-0.37	5.00	-0.37	5.00	-0.37	5.00	-0.37	-38.99	-0.15	361.54	0.00	0.49	0.15	0.51	-1.54	0.00
135.3	5.00	357.96	0.00	0.00	5.00	-2.04	5.00	-2.04	5.00	-2.04	5.00	-2.04	-38.49	-0.15	359.63	0.00	0.49	0.15	0.51	0.37	0.00
135.4	5.00	358.20	0.00	0.00	5.00	-1.80	5.00	-1.80	5.00	-1.80	5.00	-1.80	-37.99	-0.17	357.96	0.00	0.49	0.17	0.52	2.04	0.00
135.5	5.00	360.11	0.00	0.00	5.00	0.11	5.00	0.11	5.00	0.11	5.00	0.11	-37.49	-0.18	358.20	0.00	0.49	0.18	0.52	1.80	0.00
135.6	5.00	361.77	0.00	0.00	5.00	1.77	5.00	1.77	5.00	1.77	5.00	1.77	-36.99	-0.18	360.11	0.00	0.49	0.18	0.52	-0.11	0.00
135.7	5.00	361.53	0.00	0.00	5.00	1.53	5.00	1.53	5.00	1.53	5.00	1.53	-36.49	-0.17	361.77	0.00	0.49	0.17	0.52	-1.77	0.00
135.8	5.00	359.62	0.00	0.00	5.00	-0.38	5.00	-0.38	5.00	-0.38	5.00	-0.38	-35.99	-0.15	361.53	0.00	0.49	0.15	0.51	-1.53	0.00
135.9	5.00	357.97	0.00	0.00	5.00	-2.03	5.00	-2.03	5.00	-2.03	5.00	-2.03	-35.49	-0.16	359.62	0.00	0.49	0.16	0.51	0.38	0.00
136	5.00	358.22	0.00	0.00	5.00	-1.78	5.00	-1.78	5.00	-1.78	5.00	-1.78	-34.99	-0.18	357.97	0.00	0.49	0.18	0.52	2.03	0.00
136.1	5.00	360.12	0.00	0.00	5.00	0.12	5.00	0.12	5.00	0.12	5.00	0.12	-34.49	-0.19	358.22	0.00	0.49	0.19	0.53	1.78	0.00
136.2	5.00	361.76	0.00	0.00	5.00	1.76	5.00	1.76	5.00	1.76	5.00	1.76	-33.99	-0.19	360.12	0.00	0.49	0.19	0.53	-0.12	0.00
136.3	5.00	361.51	0.00	0.00	5.00	1.51	5.00	1.51	5.00	1.51	5.00	1.51	-33.49	-0.18	361.76	0.00	0.49	0.18	0.52	-1.76	0.00
136.4	5.00	359.62	0.00	0.00	5.00	-0.38	5.00	-0.38	5.00	-0.38	5.00	-0.38	-32.99	-0.16	361.51	0.00	0.49	0.16	0.52	-1.51	0.00
136.5	5.00	357.97	0.00	0.00	5.00	-2.03	5.00	-2.03	5.00	-2.03	5.00	-2.03	-32.49	-0.17	359.62	0.00	0.49	0.17	0.52	0.38	0.00
136.6	5.00	358.23	0.00	0.00	5.00	-1.77	5.00	-1.77	5.00	-1.77	5.00	-1.77	-31.99	-0.18	357.97	0.00	0.49	0.18	0.52	2.03	0.00
136.7	5.00	360.12	0.00	0.00	5.00	0.12	5.00	0.12	5.00	0.12	5.00	0.12	-31.49	-0.20	358.23	0.00	0.49	0.20	0.53	1.77	0.00
136.8	5.00	361.75	0.00	0.00	5.00	1.75	5.00	1.75	5.00	1.75	5.00	1.75	-30.99	-0.20	360.12	0.00	0.49	0.20	0.53	-0.12	0.00
136.9	5.00	361.50	0.00	0.00	5.00	1.50	5.00	1.50	5.00	1.50	5.00	1.50	-30.49	-0.18	361.75	0.00	0.49	0.18	0.52	-1.75	0.00
137	5.00	359.61	0.00	0.00	5.00	-0.39	5.00	-0.39	5.00	-0.39	5.00	-0.39	-29.99	-0.17	361.50	0.00	0.49	0.17	0.52	-1.50	0.00
137.1	5.00	357.98	0.00	0.00	5.00	-2.02	5.00	-2.02	5.00	-2.02	5.00	-2.02	-29.49	-0.17	359.61	0.00	0.49	0.17	0.52	0.39	0.00
137.2	5.00	358.24	0.00	0.00	5.00	-1.76	5.00	-1.76	5.00	-1.76	5.00	-1.76	-28.99	-0.19	357.98	0.00	0.49	0.19	0.53	2.02	0.00
137.3	5.00	360.12	0.00	0.00	5.00	0.12	5.00	0.12	5.00	0.12	5.00	0.12	-28.49	-0.21	358.24	0.00	0.49	0.21	0.53	1.76	0.00
137.4	5.00	361.74	0.00	0.00	5.00	1.74	5.00	1.74	5.00	1.74	5.00	1.74	-27.99	-0.20	360.12	0.00	0.49	0.20	0.53	-0.12	0.00
137.5	5.00	361.48	0.00	0.00	5.00	1.48	5.00	1.48	5.00	1.48	5.00	1.48	-27.49	-0.19	361.74	0.00	0.49	0.19	0.53	-1.74	0.00
137.6	5.00	359.60	0.00	0.00	5.00	-0.40	5.00	-0.40	5.00	-0.40	5.00	-0.40	-26.99	-0.18	361.48	0.00	0.49	0.18	0.52	-1.48	0.00
137.7	5.00	357.98	0.00	0.00	5.00	-2.02	5.00	-2.02	5.00	-2.02	5.00	-2.02	-26.49	-0.18	359.60	0.00	0.49	0.18	0.52	0.40	0.00
137.8	5.00	358.25	0.00	0.00	5.00	-1.75	5.00	-1.75	5.00	-1.75	5.00	-1.75	-25.99	-0.20	357.98	0.00	0.49	0.20	0.53	2.02	0.00
137.9	5.00	360.12	0.00	0.00	5.00	0.12	5.00	0.12	5.00	0.12	5.00	0.12	-25.49	-0.21	358.25	0.00	0.49	0.21	0.54	1.75	0.00
138	5.00	361.74	0.00	0.00	5.00	1.74	5.00	1.74	5.00	1.74	5.00	1.74	-24.99	-0.21	360.12	0.00	0.49	0.21	0.54	-0.12	0.00
138.1	5.00	361.47	0.00	0.00	5.00	1.47	5.00	1.47	5.00	1.47	5.00	1.47	-24.49	-0.20	361.74	0.00	0.49	0.20	0.53	-1.74	0.00
138.2	5.00	359.60	0.00	0.00	5.00	-0.40	5.00	-0.40	5.00	-0.40	5.00	-0.40	-23.99	-0.18	361.47	0.00	0.49	0.18	0.53	-1.47	0.00
138.3	5.00	357.99	0.00	0.00	5.00	-2.01	5.00	-2.01	5.00	-2.01	5.00	-2.01	-23.49	-0.19	359.60	0.00	0.49	0.19	0.53	0.40	0.00
138.4	5.00	358.26	0.00	0.00	5.00	-1.74	5.00	-1.74	5.00	-1.74	5.00	-1.74	-22.99	-0.20	357.99	0.00	0.49	0.20	0.53	2.01	0.00
138.5	5.00	360.13	0.00	0.00	5.00	0.13	5.00	0.13	5.00	0.13	5.00	0.13	-22.49	-0.22	358.26	0.00	0.49	0.22	0.54	1.74	0.00
138.6	5.00	361.73	0.00	0.00	5.00	1.73	5.00	1.73	5.00	1.73	5.00	1.73	-21.99	-0.22	360.13	0.00	0.49	0.22	0.54	-0.13	0.00
138.7	5.00	361.46	0.00	0.00	5.00	1.46	5.00	1.46	5.00	1.46	5.00	1.46	-21.49	-0.20	361.73	0.00	0.49	0.20	0.53	-1.73	0.00
138.8	5.00	359.59	0.00	0.00	5.00	-0.41	5.00	-0.41	5.00	-0.41	5.00	-0.41	-20.99	-0.19	361.46	0.00	0.49	0.19	0.53	-1.46	0.00
138.9	5.00	358.00	0.00	0.00	5.00	-2.00	5.00	-2.00	5.00	-2.00	5.00	-2.00	-20.49	-0.19	359.59	0.00	0.49	0.19	0.53	0.41	0.00
139	5.00	358.27	0.00	0.00	5.00	-1.73	5.00	-1.73	5.00	-1.73	5.00	-1.73	-20.00	-0.21	358.00	0.00	0.50	0.21	0.54	2.00	0.00
139.1	5.00	360.13	0.00	0.00	5.00	0.13	5.00	0.13	5.00	0.13	5.00	0.13	-19.50	-0.23	358.27	0.00	0.50	0.23	0.54	1.73	0.00
139.2	5.00	361.72	0.00	0.00	5.00	1.72	5.00	1.72	5.00	1.72	5.00	1.72	-19.00	-0.23	360.13	0.00	0.50	0.23	0.54	-0.13	0.00
139.3	5.00	361.44	0.00	0.00	5.00	1.44	5.00	1.44	5.00	1.44	5.00	1.44	-18.50	-0.21	361.72	0.00	0.50	0.21	0.54	-1.72	0.00
139.4	5.00	359.58	0.00	0.00	5.00	-0.42	5.00	-0.42	5.00	-0.42	5.00	-0.42	-18.00	-0.20	361.44	0.00	0.50	0.20	0.53	-1.44	0.00
139.5	5.00	358.00	0.00	0.00	5.00	-2.00	5.00	-2.00	5.00	-2.00	5.00	-2.00	-17.50	-0.20	359.58	0.00	0.50	0.20	0.54	0.42	0.00
139.6	5.00	358.28	0.00	0.00	5.00	-1.72	5.00	-1.72	5.00	-1.72	5.00	-1.72	-17.00	-0.22	358.00	0.00	0.50	0.22	0.54	2.00	0.00
139.7	5.00	360.13	0.00	0.00	5.00	0.13	5.00	0.13	5.00	0.13	5.00	0.13	-16.50	-0.23	358.28	0.00	0.50	0.23	0.55	1.72	0.00
139.8	5.00	361.71	0.00	0.00	5.00	1.71	5.00	1.71	5.00	1.71	5.00	1.71	-16.00	-0.23	360.13	0.00	0.50	0.23	0.55	-0.13	0.00
139.9	5.00	361.43	0.00	0.00	5.00	1.43	5.00	1.43	5.00	1.43	5.00	1.43	-15.50	-0.22	361.71	0.00	0.50	0.22	0.54	-1.71	0.00
140	5.00	359.58	0.00	0.00	5.00	-0.42	5.00	-0.42	5.00	-0.42	5.00	-0.42	-15.00	-0.21	361.43	0.00	0.50	0.21	0.54	-1.43	0.00
140.1	5.00	358.01	0.00	0.00	5.00	-1.99	5.00	-1.99	5.00	-1.99	5.00	-1.99	-14.50	-0.21	359.58	0.00	0.50	0.21	0.54	0.42	0.00
140.2	5.00	358.29	0.00	0.00	5.00	-1.71	5.00	-1.71	5.00	-1.71	5.00	-1.71	-14.00	-0.23	358.01	0.00	0.50	0.23	0.55	1.99	0.00
140.3	5.00	360.14	0.00	0.00	5.00	0.14	5.00	0.14	5.00	0.14	5.00	0.14	-13.50	-0.24	358.29	0.00	0.50	0.24	0.55	1.71	0.00
140.4	5.00	361.70	0.00	0.00	5.00	1.70	5.00	1.70	5.00	1.70	5.00	1.70	-13.00	-0.24	360.14	0.00	0.50	0.24	0.55	-0.14	0.00
140.5	5.00	361.42	0.00	0.00	5.00	1.42	5.00	1.42	5.00	1.42	5.00	1.42	-12.50	-0.23	361.70	0.00	0.50	0.23	0.55	-1.70	0.00
140.6	5.00	359.57	0.00	0.00	5.00	-0.43	5.00	-0.43	5.00	-0.43	5.00	-0.43	-12.00	-0.21	361.42	0.00	0.50	0.21	0.54	-1.42	0.00
140.7	5.00	358.01	0.00	0.00	5.00	-1.99	5.00	-1.99	5.00	-1.99	5.00	-1.99	-11.50	-0.22	359.57	0.00	0.50	0.22	0.54	0.43	0.00
140.8	5.00	358.30	0.00	0.00	5.00	-1.70	5.00	-1.70	5.00	-1.70	5.00	-1.70	-11.00	-0.23	358.01	0.00	0.50	0.23	0.55	1.99	0.00
140.9	5.00	360.14	0.00	0.00	5.00	0.14	5.00	0.14	5.00	0.14	5.00	0.14	-10.50	-0.25	358.30	0.00	0.50	0.25	0.56	1.70	0.00
141	5.00	361.69	0.00	0.00	5.00	1.69	5.00	1.69	5.00	1.69	5.00	1.69	-10.00	-0.25	360.14	0.00	0.50	0.25	0.56	-0.14	0.00
141.1	5.00	361.40	0.00	0.00	5.00	1.40	5.00	1.40	5.00	1.40	5.00	1.40	-9.50	-0.23	361.69	0.00	0.50	0.23	0.55	-1.69	0.00
141.2	5.00	359.56	0.00	0.00	5.00	-0.44	5.00	-0.44	5.00	-0.44	5.00	-0.44	-9.00	-0.22	361.40	0.00	0.50	0.22	0.55	-1.40	0.00
141.3	5.00	358.02	0.00	0.00	5.00	-1.98	5.00	-1.98	5.00	-1.98	5.00	-1.98	-8.50	-0.23	359.56	0.00	0.50	0.23	0.55	0.44	0.00
141.4	5.00	358.31	0.00	0.00	5.00	-1.69	5.00	-1.69	5.00	-1.69	5.00	-1.69	-8.00	-0.24	358.02	0.00	0.50	0.24	0.55	1.98	0.00
141.5	5.00	360.14	0.00	0.00	5.00	0.14	5.00	0.14	5.00	0.14	5.00	0.14	-7.50	-0.26	358.31	0.00	0.50	0.26	0.56	1.69	0.00
141.6	5.00	361.68	0.00	0.00	5.00	1.68	5.00	1.68	5.00	1.68	5.00	1.68	-7.00	-0.26	360.14	0.00	0.50	0.26	0.56	-0.14	0.00
141.7	5.00	361.39	0.00	0.00	5.00	1.39	5.00	1.39	5.00	1.39	5.00	1.39	-6.50	-0.24	361.68	0.00	0.50	0.24	0.55	-1.68	0.00
141.8	5.00	359.56	0.00	0.00	5.00	-0.44	5.00	-0.44	5.00	-0.44	5.00	-0.44	-6.00	-0.23	361.39	0.00	0.50	0.23	0.55	-1.39	0.00
141.9	5.00	358.03	0.00	0.00	5.00	-1.97	5.00	-1.97	5.00	-1.97	5.00	-1.97	-5.50	-0.23	359.56	0.00	0.50	0.23	0.55	0.44	0.00
142	5.00	358.32	0.00	0.00	5.00	-1.68	5.00	-1.68	5.00	-1.68	5.00	-1.68	-5.00	-0.25	358.03	0.00	0.50	0.25	0.56	1.97	0.00
142.1	5.00	360.14	0.00	0.00	5.00	0.14	5.00	0.14	5.00	0.14	5.00	0.14	-4.50	-0.26	358.32	0.00	0.50	0.26	0.57	1.68	0.00
142.2	5.00	361.67	0.00	0.00	5.00	1.67	5.00	1.67	5.00	1.67	5.00	1.67	-4.00	-0.26	360.14	0.00	0.50	0.26	0.57	-0.14	0.00
142.3	5.00	361.37	0.00	0.00	5.00	1.37	5.00	1.37	5.00	1.37	5.00	1.37	-3.50	-0.25	361.67	0.00	0.50	0.25	0.56	-1.67	0.00
142.4	5.00	359.55	0.00	0.00	5.00	-0.45	5.00	-0.45	5.00	-0.45	5.00	-0.45	-3.00	-0.24	361.37	0.00	0.50	0.24	0.55	-1.37	0.00
142.5	5.00	358.03	0.00	0.00	5.00	-1.97	5.00	-1.97	5.00	-1.97	5.00	-1.97	-2.50	-0.24	359.55	0.00	0.50	0.24	0.56	0.45	0.00

142.6	5.00	358.33	0.00	0.00	5.00	-1.67	5.00	-1.67	5.00	-1.67	5.00	-1.67	-2.00	-0.26	358.03	0.00	0.50	0.26	0.56	1.97	0.00
142.7	5.00	360.15	0.00	0.00	5.00	0.15	5.00	0.15	5.00	0.15	5.00	0.15	-1.50	-0.27	358.33	0.00	0.50	0.27	0.57	1.67	0.00
142.8	5.00	361.66	0.00	0.00	5.00	1.66	5.00	1.66	5.00	1.66	5.00	1.66	-1.00	-0.27	360.15	0.00	0.50	0.27	0.57	-0.15	0.00
142.9	5.00	361.36	0.00	0.00	5.00	1.36	5.00	1.36	5.00	1.36	5.00	1.36	-0.50	-0.26	361.66	0.00	0.50	0.26	0.56	-1.66	0.00

Appendix 2: Mobility Code from 2016 Rovers

```
#include <ros/ros.h>

//ROS libraries
#include <angles/angles.h>
#include <random_numbers/random_numbers.h>
#include <tf/transform_datatypes.h>

//ROS messages
#include <std_msgs/Int16.h>
#include <std_msgs/UInt8.h>
#include <std_msgs/String.h>
#include <sensor_msgs/Joy.h>
#include <sensor_msgs/Range.h>
#include <geometry_msgs/Pose2D.h>
#include <geometry_msgs/Twist.h>
#include <nav_msgs/Odometry.h>

//Custom messages
#include <shared_messages/TagsImage.h>

// To handle shutdown signals so the node quits properly in response to "rosnode kill"
#include <ros/ros.h>
#include <signal.h>

using namespace std;

//Random number generator
random_numbers::RandomNumberGenerator* rng;

//Mobility Logic Functions
void setVelocity(double linearVel, double angularVel);

//Numeric Variables
geometry_msgs::Pose2D currentLocation;
geometry_msgs::Pose2D goalLocation;
int currentMode = 0;
float mobilityLoopTimeStep = 0.1; //time between the mobility loop calls
float status_publish_interval = 5;
float killSwitchTimeout = 10;
std_msgs::Int16 targetDetected; //ID of the detected target
bool targetsCollected [512] = {0}; //array of booleans indicating whether each target ID has been found
int initiate = 0;
int obstacle = 0;
int stage = 0;
int nest = 0;
int competitionround = 0;
float ygoal = 0.0;
int rake = 0;
```

```
int tag = 0;
int recal = 0;

// state machine states
#define STATE_MACHINE_INIT       0
#define STATE_MACHINE_RECAL    1
#define STATE_MACHINE_RUNWAY            2
#define STATE_MACHINE_SEEK      3
#define STATE_MACHINE_SWEEP   4
#define STATE_MACHINE_RETURN 5
int stateMachineState = STATE_MACHINE_INIT;

geometry_msgs::Twist velocity;
char host[128];
string publishedName;
char prev_state_machine[128];

//Publishers
ros::Publisher velocityPublish;
ros::Publisher stateMachinePublish;
ros::Publisher status_publisher;
ros::Publisher targetCollectedPublish;
ros::Publisher targetPickUpPublish;
ros::Publisher targetDropOffPublish;

//Subscribers
ros::Subscriber joySubscriber;
ros::Subscriber modeSubscriber;
ros::Subscriber targetSubscriber;
ros::Subscriber obstacleSubscriber;
ros::Subscriber odometrySubscriber;
ros::Subscriber targetsCollectedSubscriber;

//Timers
ros::Timer stateMachineTimer;
ros::Timer publish_status_timer;
ros::Timer killSwitchTimer;

// OS Signal Handler
void sigintEventHandler(int signal);

//Callback handlers
void joyCmdHandler(const geometry_msgs::Twist::ConstPtr& message);
void modeHandler(const std_msgs::UInt8::ConstPtr& message);
void targetHandler(const shared_messages::TagsImage::ConstPtr& tagInfo);
void obstacleHandler(const std_msgs::UInt8::ConstPtr& message);
void odometryHandler(const nav_msgs::Odometry::ConstPtr& message);
void mobilityStateMachine(const ros::TimerEvent&);
void publishStatusTimerEventHandler(const ros::TimerEvent& event);
void targetsCollectedHandler(const std_msgs::Int16::ConstPtr& message);
void killSwitchTimerEventHandler(const ros::TimerEvent& event);
```

```
int main(int argc, char **argv) {

   gethostname(host, sizeof (host));
   string hostname(host);

   rng = new random_numbers::RandomNumberGenerator(); //instantiate random number generator
            goalLocation.theta = rng->uniformInteger(0, 1);
            if (goalLocation.theta == 0) {
                     goalLocation.theta = 0.5 * M_PI;
            }
            else {
                     goalLocation.theta = 1.5 * M_PI;
            }

   targetDetected.data = -1; //initialize target detected



   if (argc >= 2) {
      publishedName = argv[1];
      cout << "Welcome to the world of tomorrow " << publishedName << "!  Mobility module started." <<
endl;
   } else {
      publishedName = hostname;
      cout << "No Name Selected. Default is: " << publishedName << endl;
   }

   // NoSignalHandler so we can catch SIGINT ourselves and shutdown the node
   ros::init(argc, argv, (publishedName + "_MOBILITY"), ros::init_options::NoSigintHandler);
   ros::NodeHandle mNH;

   signal(SIGINT, sigintEventHandler); // Register the SIGINT event handler so the node can shutdown
properly

   joySubscriber = mNH.subscribe((publishedName + "/joystick"), 10, joyCmdHandler);
   modeSubscriber = mNH.subscribe((publishedName + "/mode"), 1, modeHandler);
   targetSubscriber = mNH.subscribe((publishedName + "/targets"), 10, targetHandler);
   obstacleSubscriber = mNH.subscribe((publishedName + "/obstacle"), 10, obstacleHandler);
   odometrySubscriber = mNH.subscribe((publishedName + "/odom/ekf"), 10, odometryHandler);
   targetsCollectedSubscriber = mNH.subscribe(("targetsCollected"), 10, targetsCollectedHandler);

   status_publisher = mNH.advertise<std_msgs::String>((publishedName + "/status"), 1, true);
   velocityPublish = mNH.advertise<geometry_msgs::Twist>((publishedName + "/velocity"), 10);
   stateMachinePublish = mNH.advertise<std_msgs::String>((publishedName + "/state_machine"), 1,
true);
   targetCollectedPublish = mNH.advertise<std_msgs::Int16>(("targetsCollected"), 1, true);
   targetPickUpPublish = mNH.advertise<sensor_msgs::Image>((publishedName + "/targetPickUpImage"),
1, true);
   targetDropOffPublish = mNH.advertise<sensor_msgs::Image>((publishedName +
"/targetDropOffImage"), 1, true);
```

```
  publish_status_timer = mNH.createTimer(ros::Duration(status_publish_interval),
publishStatusTimerEventHandler);
  //killSwitchTimer = mNH.createTimer(ros::Duration(killSwitchTimeout), killSwitchTimerEventHandler);
  stateMachineTimer = mNH.createTimer(ros::Duration(mobilityLoopTimeStep), mobilityStateMachine);

  ros::spin();

  return EXIT_SUCCESS;
}

void mobilityStateMachine(const ros::TimerEvent&) {
  std_msgs::String stateMachineMsg;

  if (currentMode == 2 || currentMode == 3) { //Robot is in automode

                switch(stateMachineState) {

                        case STATE_MACHINE_INIT: {
                        stateMachineMsg.data = "INIT";
                                if (angles::shortest_angular_distance(currentLocation.theta,
goalLocation.theta) > 0.1) {
                                        setVelocity(0.0, 0.2); //rotate left
                                }
                                else if (angles::shortest_angular_distance(currentLocation.theta,
goalLocation.theta) < -0.1) {
                                        setVelocity(0.0, -0.2); //rotate right
                                }
                                else if (obstacle == 4) {
                                        if ((currentLocation.y > 9.0) || (currentLocation.y < -9.0)) {
                                                if (currentLocation.y > 9.0) {
                                                        competitionround = 2;
                                                        setVelocity(0.0, 0.0);
                                                        ygoal = 9.9;  //calibrate!!!
                                                        stateMachineState =
STATE_MACHINE_SWEEP;
                                                }
                                                else {
                                                        competitionround = 2;
                                                        setVelocity(0.0, 0.0);
                                                        ygoal = -9.9;  //calibrate!!!
                                                        stateMachineState =
STATE_MACHINE_SWEEP;
                                                }
                                        }
                                        else {
                                                if (currentLocation.y > 0) {
                                                        competitionround = 1;
                                                        setVelocity(0.0, 0.0);
                                                        ygoal = 6.4;  //calibrate!!!
```

```
                                                stateMachineState =
STATE_MACHINE_SWEEP;
                                        }
                                        else {
                                                competitionround = 1;
                                                setVelocity(0.0, 0.0);
                                                ygoal = -6.4;  //calibrate!!!
                                                stateMachineState =
STATE_MACHINE_SWEEP;
                                        }
                                }
                        }
                        else if (obstacle == 5) {
                                setVelocity(0.2, 0.0);
                        }
                        else {
                                setVelocity(0.3, 0.0);
                        }
                   break;
                }

                case STATE_MACHINE_RECAL: { //recalibration
                stateMachineMsg.data = "STAGE0";
                        recal = recal + 1;
                        if (recal > 10) {
                                recal = 0;
                                if (competitionround == 2) {
                                        competitionround = 1;
                                }
                                else {
                                        competitionround = 2;
                                }
                        }
                        else {
                                if (ygoal > 0) {
                                        if
(angles::shortest_angular_distance(currentLocation.theta, 0.5 * M_PI) > 0.1) {
                                                setVelocity(0.0, 0.2); //rotate left
                                        }
                                        else if
(angles::shortest_angular_distance(currentLocation.theta, 0.5 * M_PI) < -0.1) {
                                                setVelocity(0.0, -0.2); //rotate right
                                        }
                                        else if (obstacle == 3) {
                                                setVelocity(-0.1, 0.0);
                                        }
                                        else if (obstacle == 4) {
                                                if (competitionround == 2) {
                                                        ygoal = 9.9; //calibrate!!!
                                                }
                                                else {
```

```
                                                ygoal = 6.4;  //calibrate!!!
                                        }
                                        setVelocity(0.0, 0.0);
                                        stateMachineState =
STATE_MACHINE_SWEEP;
                                }
                                else if (obstacle == 5) {
                                        setVelocity(0.1, 0.0);
                                }
                                else {
                                        setVelocity(0.3, 0.0);
                                }
                        }
                        else {
                                if
(angles::shortest_angular_distance(currentLocation.theta, 1.5 * M_PI) > 0.1) {
                                        setVelocity(0.0, 0.2); //rotate left
                                }
                                else if
(angles::shortest_angular_distance(currentLocation.theta, 1.5 * M_PI) < -0.1) {
                                        setVelocity(0.0, -0.2); //rotate right
                                }
                                else if (obstacle == 3) {
                                        setVelocity(-0.1, 0.0);
                                }
                                else if (obstacle == 4) {
                                        if (competitionround == 2) {
                                                ygoal = -9.9;  //calibrate!!!
                                        }
                                        else {
                                                ygoal = -6.4;  //calibrate!!!
                                        }
                                        setVelocity(0.0, 0.0);
                                        stateMachineState =
STATE_MACHINE_SWEEP;
                                }
                                else if (obstacle == 5) {
                                        setVelocity(0.2, 0.0);
                                }
                                else {
                                        setVelocity(0.3, 0.0);
                                }
                        }
                }
           break;
        }

        case STATE_MACHINE_RUNWAY: { //east to west sweep across nest
        stateMachineMsg.data = "STAGE1";
                if (angles::shortest_angular_distance(currentLocation.theta, M_PI) >
0.1) {
```

```
                                        setVelocity(0.0, 0.2); //rotate left
                                }
                                else if (angles::shortest_angular_distance(currentLocation.theta,
M_PI) < -0.1) {
                                        setVelocity(0.0, -0.2); //rotate right
                                }
                                else if (obstacle == 3) {
                                        setVelocity(-0.1, 0.0);
                                }
                                else if (obstacle == 4) {
                                                if (nest != 0) {
                                                        setVelocity(0.0, 0.0);
                                                        stateMachineState =
STATE_MACHINE_RECAL;
                                                }
                                                else if (tag == 1) { //found tag so repeat loop
                                                        ygoal = rake/2;
                                                        goalLocation.y = currentLocation.y + ygoal;
                                                        tag = 0;
                                                        setVelocity(0.0, 0.0);
                                                        stateMachineState =
STATE_MACHINE_SEEK;
                                                }
                                                else { // not tag found pick new loop
                                                        if (competitionround == 2) {
                                                                rake = rng->uniformInteger(-21,
21);
                                                                if (rake == 0) {
                                                                        rake = 2;
                                                                }
                                                                ygoal = rake/2;
                                                                goalLocation.y = currentLocation.y
+ ygoal;
                                                                setVelocity(0.0, 0.0);
                                                                stateMachineState =
STATE_MACHINE_SEEK;
                                                        }
                                                        else {
                                                                rake = rng->uniformInteger(-14,
14);
                                                                if (rake == 0) {
                                                                        rake = 2;
                                                                }
                                                                ygoal = rake/2;
                                                                goalLocation.y = currentLocation.y
+ ygoal;
                                                                setVelocity(0.0, 0.0);
                                                                stateMachineState =
STATE_MACHINE_SEEK;
                                                        }
                                                }
```

```
                        }
                        else if (obstacle == 5) {
                                setVelocity(0.1, 0.0);
                        }
                        else {
                                setVelocity(0.3, 0.0);
                        }
                   break;
                }

                case STATE_MACHINE_SEEK: { //seek y position for rake
                stateMachineMsg.data = "STAGE2";
                        if (ygoal > 0) {
                                if (angles::shortest_angular_distance(currentLocation.theta,
0.5 * M_PI) > 0.1) {
                                        setVelocity(0.0, 0.2); //rotate left
                                }
                                else if
(angles::shortest_angular_distance(currentLocation.theta, 0.5 * M_PI) < -0.1) {
                                        setVelocity(0.0, -0.2); //rotate right
                                }
                                else if ((obstacle == 1) || (obstacle == 2) || (obstacle == 3)) {
                                        setVelocity(0.0, 0.0);
                                        stateMachineState = STATE_MACHINE_SWEEP;
                                }
                                else if (currentLocation.y > goalLocation.y) {
                                        setVelocity(0.0, 0.0);
                                        stateMachineState = STATE_MACHINE_SWEEP;
                                }
                                else if (obstacle == 5) {
                                        setVelocity(0.1, 0.0);
                                }
                                else {
                                        setVelocity(0.3, 0.0);
                                }
                        }
                        else {
                                if (angles::shortest_angular_distance(currentLocation.theta,
1.5 * M_PI) > 0.1) {
                                        setVelocity(0.0, 0.2); //rotate left
                                }
                                else if
(angles::shortest_angular_distance(currentLocation.theta, 1.5 * M_PI) < -0.1) {
                                        setVelocity(0.0, -0.2); //rotate right
                                }
                                else if ((obstacle == 1) || (obstacle == 2) || (obstacle == 3)) {
                                        setVelocity(0.0, 0.0);
                                        stateMachineState = STATE_MACHINE_SWEEP;
                                }
                                else if (currentLocation.y < goalLocation.y) {
                                        setVelocity(0.0, 0.0);
```

```
                                        stateMachineState = STATE_MACHINE_SWEEP;
                                }
                                else if (obstacle == 5) {
                                        setVelocity(0.1, 0.0);
                                }
                                else {
                                        setVelocity(0.3, 0.0);
                                }
                        }
                  break;
                }

                case STATE_MACHINE_SWEEP: { //west to east sweep looking for tags
                stateMachineMsg.data = "STAGE3";
                        goalLocation.theta = 0;
                        if (angles::shortest_angular_distance(currentLocation.theta,
goalLocation.theta) > 0.1) {
                                setVelocity(0.0, 0.2); //rotate left
                        }
                        else if (angles::shortest_angular_distance(currentLocation.theta,
goalLocation.theta) < -0.1) {
                                setVelocity(0.0, -0.2); //rotate right
                        }
                        else if (obstacle == 3) {
                                setVelocity(-0.1, 0.0);
                        }
                        else if (obstacle == 4) {
                                goalLocation.y = currentLocation.y - ygoal;
                                setVelocity(0.0, 0.0);
                                stateMachineState = STATE_MACHINE_RETURN;
                        }
                        else if (tag == 1) {
                                setVelocity(0.5, 0.0);
                        }
                        else if (obstacle == 5) {
                                setVelocity(0.1, 0.0);
                        }
                        else {
                                setVelocity(0.3, 0.0);
                        }
                  break;
                }

                case STATE_MACHINE_RETURN: { //seek y position for nest
                stateMachineMsg.data = "STAGE4";
                        if (ygoal < 0) {
                                if (angles::shortest_angular_distance(currentLocation.theta,
0.5 * M_PI) > 0.1) {
                                        setVelocity(0.0, 0.2); //rotate left
                                }
```

```
                                        else if
(angles::shortest_angular_distance(currentLocation.theta, 0.5 * M_PI) < -0.1) {
                                                setVelocity(0.0, -0.2); //rotate right
                                        }
                                        else if ((obstacle == 1) || (obstacle == 2) || (obstacle == 3)) {
                                                setVelocity(0.0, 0.0);
                                                stateMachineState = STATE_MACHINE_RUNWAY;
                                        }
                                        else if (currentLocation.y > goalLocation.y) {
                                                setVelocity(0.0, 0.0);
                                                nest = 1;
                                                stateMachineState = STATE_MACHINE_RUNWAY;
                                        }
                                        else if (obstacle == 5) {
                                                setVelocity(0.1, 0.0);
                                        }
                                        else {
                                                setVelocity(0.3, 0.0);
                                        }
                                }
                                else {
                                        if (angles::shortest_angular_distance(currentLocation.theta,
1.5 * M_PI) > 0.1) {
                                                setVelocity(0.0, 0.2); //rotate left
                                        }
                                        else if
(angles::shortest_angular_distance(currentLocation.theta, 1.5 * M_PI) < -0.1) {
                                                setVelocity(0.0, -0.2); //rotate right
                                        }
                                        else if ((obstacle == 1) || (obstacle == 2) || (obstacle == 3)) {
                                                setVelocity(0.0, 0.0);
                                                stateMachineState = STATE_MACHINE_RUNWAY;
                                        }
                                        else if (currentLocation.y < goalLocation.y) {
                                                setVelocity(0.0, 0.0);
                                                nest = 1;
                                                stateMachineState = STATE_MACHINE_RUNWAY;
                                        }
                                        else if (obstacle == 5) {
                                                setVelocity(0.1, 0.0);
                                        }
                                        else {
                                                setVelocity(0.3, 0.0);
                                        }
                                }
                        }
                        default: {
                           break;
                        }
                }
        }
```

```
    else { // mode is NOT auto

      // publish current state for the operator to see
      stateMachineMsg.data = "WAITING";
    }

    // publish state machine string for user, only if it has changed, though
    if (strcmp(stateMachineMsg.data.c_str(), prev_state_machine) != 0) {
      stateMachinePublish.publish(stateMachineMsg);
      sprintf(prev_state_machine, "%s", stateMachineMsg.data.c_str());
    }
}

void setVelocity(double linearVel, double angularVel)
{
  // Stopping and starting the timer causes it to start counting from 0 again.
  // As long as this is called before the kill swith timer reaches killSwitchTimeout seconds
  // the rover's kill switch wont be called.
  killSwitchTimer.stop();
  killSwitchTimer.start();

  velocity.linear.x = linearVel * 1.5;
  velocity.angular.z = angularVel * 8; //scaling factor for sim; removed by aBridge node
  velocityPublish.publish(velocity);
}

/************************
 * ROS CALLBACK HANDLERS
 ************************/

void targetHandler(const shared_messages::TagsImage::ConstPtr& message) {

        //if this is the goal target
        if (message->tags.data[0] == 256) {
                nest = 0;
                recal = 0;
                //if we were returning with a target
          if (targetDetected.data != -1) {
                        //publish to scoring code
                        targetDropOffPublish.publish(message->image);
                        targetDetected.data = -1;
                        tag = 0;
          }
        }

        //if target has not previously been detected
        else if (targetDetected.data == -1) {

        //check if target has not yet been collected
        if (!targetsCollected[message->tags.data[0]]) {
```

```
                        //copy target ID to class variable
                        targetDetected.data = message->tags.data[0];

                        //publish detected target
                        targetCollectedPublish.publish(targetDetected);

                        //publish to scoring code
                        targetPickUpPublish.publish(message->image);

                        //switch to transform state to trigger return to center
                        tag = 1;
                }
    }
}

void modeHandler(const std_msgs::UInt8::ConstPtr& message) {
        currentMode = message->data;
        setVelocity(0.0, 0.0);
}

void obstacleHandler(const std_msgs::UInt8::ConstPtr& message) {
        if (message->data > 0) {
                if (message->data == 1) {
                        obstacle = 1;
                }
                else if (message->data == 2) {
                        obstacle = 2;
                }
                else if (message->data == 3) {
                        obstacle = 3;
                }
                else if (message->data == 4) {
                        obstacle = 4;
                }
                else if (message->data == 5) {
                        obstacle = 5;
                }
        }
        else {
                obstacle = 0;
        }
}

void odometryHandler(const nav_msgs::Odometry::ConstPtr& message) {
        //Get (x,y) location directly from pose
        currentLocation.x = message->pose.pose.position.x;
        currentLocation.y = message->pose.pose.position.y;

        //Get theta rotation by converting quaternion orientation to pitch/roll/yaw
        tf::Quaternion q(message->pose.pose.orientation.x, message->pose.pose.orientation.y,
message->pose.pose.orientation.z, message->pose.pose.orientation.w);
```

```
        tf::Matrix3x3 m(q);
        double roll, pitch, yaw;
        m.getRPY(roll, pitch, yaw);
        currentLocation.theta = yaw;
}

void joyCmdHandler(const geometry_msgs::Twist::ConstPtr& message) {
   if (currentMode == 0 || currentMode == 1)
    {
        setVelocity(message->linear.x, message->angular.z);
    }
}


void publishStatusTimerEventHandler(const ros::TimerEvent&)
{
 std_msgs::String msg;
 msg.data = "FIU Panther Swarm";
 status_publisher.publish(msg);
}

// Safety precaution. No movement commands - might have lost contact with ROS. Stop the rover.
// Also might no longer be receiving manual movement commands so stop the rover.
void killSwitchTimerEventHandler(const ros::TimerEvent& t)
{
 // No movement commands for killSwitchTime seconds so stop the rover
 setVelocity(0,0);
 double current_time = ros::Time::now().toSec();
 ROS_INFO("In mobility.cpp:: killSwitchTimerEventHander(): Movement input timeout. Stopping the rover
at %6.4f.", current_time);
}

void targetsCollectedHandler(const std_msgs::Int16::ConstPtr& message) {
        targetsCollected[message->data] = 1;
}

void sigintEventHandler(int sig)
{
   // All the default sigint handler does is call shutdown()
   ros::shutdown();
}
```

Appendix 3: Mobility Code from 2017 Rovers

```
#include <ros/ros.h>

// ROS libraries
#include <angles/angles.h>
#include <random_numbers/random_numbers.h>
#include <tf/transform_datatypes.h>
#include <tf/transform_listener.h>

// ROS messages
#include <std_msgs/Float32.h>
#include <std_msgs/Int16.h>
#include <std_msgs/UInt8.h>
#include <std_msgs/String.h>
#include <sensor_msgs/Joy.h>
#include <sensor_msgs/Range.h>
#include <geometry_msgs/Pose2D.h>
#include <geometry_msgs/Twist.h>
#include <nav_msgs/Odometry.h>
#include <apriltags_ros/AprilTagDetectionArray.h>

// Include Controllers
#include "PickUpController.h"
#include "DropOffController.h"
#include "SearchController.h"

// To handle shutdown signals so the node quits
// properly in response to "rosnode kill"
#include <ros/ros.h>
#include <signal.h>


using namespace std;

// Random number generator
random_numbers::RandomNumberGenerator* rng;

// Create controllers
PickUpController pickUpController;
DropOffController dropOffController;
SearchController searchController;

// Mobility Logic Functions
void sendDriveCommand(double linearVel, double angularVel);
void openFingers(); // Open fingers to 90 degrees
void closeFingers();// Close fingers to 0 degrees
void raiseWrist();  // Return wrist back to 0 degrees
void lowerWrist();  // Lower wrist to 50 degrees
void mapAverage();  // constantly averages last 100 positions from map
```

```
// Numeric Variables for rover positioning
geometry_msgs::Pose2D currentLocation;
geometry_msgs::Pose2D currentLocationMap;
geometry_msgs::Pose2D currentLocationAverage;
geometry_msgs::Pose2D goalLocation;

geometry_msgs::Pose2D centerLocation;
geometry_msgs::Pose2D centerLocationMap;
geometry_msgs::Pose2D centerLocationOdom;

int currentMode = 0;
float mobilityLoopTimeStep = 0.1; // time between the mobility loop calls
float status_publish_interval = 1;
float killSwitchTimeout = 10;
int xaxis = 0; //values from -15 to 15, 0 being neutral axis
int yaxis = 0;
int newspace = 1; //counter to tell if rovers first time to a goal space.
int ultgoalx = 0; //end point for a desired path
int ultgoaly = 0;
int intgoalx = 0; //next stepping point for path
int intgoaly = 0;
int unclearedspacex = 0;
int unclearedspacey = 0; //assignment for a space not gone to yet, will result in a spin at the space
int clearedspacex = 0;
int clearedspacey = 0;
int curlocx = 0; //current location in cell form
int curlocy = 0;
int zone = 0; //break down arena into zones
int zonereset = 0;
int resetstep = 1;
int turn = 0;
int step = 0;
int returntarget = 0;
int reset = 0;
int inittheta = 0;
int compround = 2;
int wall = 0;
int cubereturn = 0;
int lastcube = 0;
bool targetDetected = false;
bool targetCollected = false;

// Set true when the target block is less than targetDist so we continue
// attempting to pick it up rather than switching to another block in view.
bool lockTarget = false;

// Failsafe state. No legitimate behavior state. If in this state for too long
// return to searching as default behavior.
bool timeOut = false;

// Set to true when the center ultrasound reads less than 0.14m. Usually means
```

```
// a picked up cube is in the way.
bool blockBlock = false;

// central collection point has been seen (aka the nest)
bool centerSeen = false;

// Set true when we are insie the center circle and we need to drop the block,
// back out, and reset the boolean cascade.
bool reachedCollectionPoint = false;

// used for calling code once but not in main
bool init = false;

// used to remember place in mapAverage array
int mapCount = 0;

// How many points to use in calculating the map average position
const unsigned int mapHistorySize = 500;

// An array in which to store map positions
geometry_msgs::Pose2D mapLocation[mapHistorySize];

bool avoidingObstacle = false;

float searchVelocity = 0.2; // meters/second

std_msgs::String msg;

// state machine states
#define STATE_MACHINE_TRANSFORM 0
#define STATE_MACHINE_ROTATE 1
#define STATE_MACHINE_SKID_STEER 2
#define STATE_MACHINE_PICKUP 3
#define STATE_MACHINE_DROPOFF 4

int stateMachineState = STATE_MACHINE_TRANSFORM;

geometry_msgs::Twist velocity;
char host[128];
string publishedName;
char prev_state_machine[128];

// Publishers
ros::Publisher stateMachinePublish;
ros::Publisher status_publisher;
ros::Publisher fingerAnglePublish;
ros::Publisher wristAnglePublish;
ros::Publisher infoLogPublisher;
ros::Publisher driveControlPublish;

// Subscribers
```

```
ros::Subscriber joySubscriber;
ros::Subscriber modeSubscriber;
ros::Subscriber targetSubscriber;
ros::Subscriber obstacleSubscriber;
ros::Subscriber odometrySubscriber;
ros::Subscriber mapSubscriber;


// Timers
ros::Timer stateMachineTimer;
ros::Timer publish_status_timer;
ros::Timer targetDetectedTimer;

// records time for delays in sequanced actions, 1 second resolution.
time_t timerStartTime;

// An initial delay to allow the rover to gather enough position data to
// average its location.
unsigned int startDelayInSeconds = 1;
float timerTimeElapsed = 0;

//Transforms
tf::TransformListener *tfListener;

// OS Signal Handler
void sigintEventHandler(int signal);

//Callback handlers
void joyCmdHandler(const sensor_msgs::Joy::ConstPtr& message);
void modeHandler(const std_msgs::UInt8::ConstPtr& message);
void targetHandler(const apriltags_ros::AprilTagDetectionArray::ConstPtr& tagInfo);
void obstacleHandler(const std_msgs::UInt8::ConstPtr& message);
void odometryHandler(const nav_msgs::Odometry::ConstPtr& message);
void mapHandler(const nav_msgs::Odometry::ConstPtr& message);
void mobilityStateMachine(const ros::TimerEvent&);
void publishStatusTimerEventHandler(const ros::TimerEvent& event);
void targetDetectedReset(const ros::TimerEvent& event);

int main(int argc, char **argv) {

   gethostname(host, sizeof (host));
   string hostname(host);

   // instantiate random number generator
   rng = new random_numbers::RandomNumberGenerator();

   //set initial random heading
   goalLocation.theta = rng->uniformReal(0, 2 * M_PI);

   //select initial search position 50 cm from center (0,0)
   goalLocation.x = 0.5 * cos(goalLocation.theta+M_PI);
```

```
  goalLocation.y = 0.5 * sin(goalLocation.theta+M_PI);

  centerLocation.x = 0;
  centerLocation.y = 0;
  centerLocationOdom.x = 0;
  centerLocationOdom.y = 0;

std::cout << "Hello world" << "  " << resetstep << " " << zonereset << std::endl;

  for (int i = 0; i < 100; i++) {
    mapLocation[i].x = 0;
    mapLocation[i].y = 0;
    mapLocation[i].theta = 0;
  }

  if (argc >= 2) {
    publishedName = argv[1];
    cout << "Welcome to the world of tomorrow " << publishedName
      << "!  Mobility turnDirectionule started." << endl;
  } else {
    publishedName = hostname;
    cout << "No Name Selected. Default is: " << publishedName << endl;
  }

  // NoSignalHandler so we can catch SIGINT ourselves and shutdown the node
  ros::init(argc, argv, (publishedName + "_MOBILITY"), ros::init_options::NoSigintHandler);
  ros::NodeHandle mNH;

  // Register the SIGINT event handler so the node can shutdown properly
  signal(SIGINT, sigintEventHandler);

  joySubscriber = mNH.subscribe((publishedName + "/joystick"), 10, joyCmdHandler);
  modeSubscriber = mNH.subscribe((publishedName + "/mode"), 1, modeHandler);
  targetSubscriber = mNH.subscribe((publishedName + "/targets"), 10, targetHandler);
  obstacleSubscriber = mNH.subscribe((publishedName + "/obstacle"), 10, obstacleHandler);
  odometrySubscriber = mNH.subscribe((publishedName + "/odom/filtered"), 10, odometryHandler);
  mapSubscriber = mNH.subscribe((publishedName + "/odom/ekf"), 10, mapHandler);

  status_publisher = mNH.advertise<std_msgs::String>((publishedName + "/status"), 1, true);
  stateMachinePublish = mNH.advertise<std_msgs::String>((publishedName + "/state_machine"), 1,
true);
  fingerAnglePublish = mNH.advertise<std_msgs::Float32>((publishedName + "/fingerAngle/cmd"), 1,
true);
  wristAnglePublish = mNH.advertise<std_msgs::Float32>((publishedName + "/wristAngle/cmd"), 1,
true);
  infoLogPublisher = mNH.advertise<std_msgs::String>("/infoLog", 1, true);
  driveControlPublish = mNH.advertise<geometry_msgs::Twist>((publishedName + "/driveControl"), 10);

  publish_status_timer = mNH.createTimer(ros::Duration(status_publish_interval),
publishStatusTimerEventHandler);
  stateMachineTimer = mNH.createTimer(ros::Duration(mobilityLoopTimeStep), mobilityStateMachine);
```

```
    targetDetectedTimer = mNH.createTimer(ros::Duration(0), targetDetectedReset, true);

    tfListener = new tf::TransformListener();
    std_msgs::String msg;
    msg.data = "Log Started";
    infoLogPublisher.publish(msg);

    stringstream ss;
    ss << "Rover start delay set to " << startDelayInSeconds << " seconds";
    msg.data = ss.str();
    infoLogPublisher.publish(msg);

    timerStartTime = time(0);

    ros::spin();

    return EXIT_SUCCESS;
}


// This is the top-most logic control block organised as a state machine.
// This function calls the dropOff, pickUp, and search controllers.
// This block passes the goal location to the proportional-integral-derivative
// controllers in the abridge package.
void mobilityStateMachine(const ros::TimerEvent&) {

    std_msgs::String stateMachineMsg;
    float rotateOnlyAngleTolerance = 0.4;
    int returnToSearchDelay = 5;

    // calls the averaging function, also responsible for
    // transform from Map frame to odom frame.
    mapAverage();

    // Robot is in automode
    if (currentMode == 2 || currentMode == 3) {


        // time since timerStartTime was set to current time
        timerTimeElapsed = time(0) - timerStartTime;

        // init code goes here. (code that runs only once at start of
        // auto mode but wont work in main goes here)
        if (!init) {
            if (timerTimeElapsed > startDelayInSeconds) {
                // Set the location of the center circle location in the map
                // frame based upon our current average location on the map.
                centerLocationMap.x = currentLocationAverage.x;
                centerLocationMap.y = currentLocationAverage.y;
                centerLocationMap.theta = currentLocationAverage.theta;
```

```
		//uses initial orentation to assign a number to the rover and a grid space based upon how it is orientated with collection zone

		float inittheta = angles::shortest_angular_distance(currentLocation.theta, 0);

		//give an ult goal that assumes the larger map
		if (abs(inittheta) < (22.5/180)*M_PI_2) { //zone 5
			zone = 5;
			curlocx = 10;
			curlocy = 11;
			ultgoalx = 1;
			ultgoaly = 20;
		}
		else if (inittheta > (22.5/180)*M_PI_2 && inittheta < (67.5/180)*M_PI_2) {
			zone = 6;
			curlocx = 10;
			curlocy = 10;
			ultgoalx = 1;
			ultgoaly = 10;
		}
		else if (inittheta > (67.5/180)*M_PI_2 && inittheta < (112.5/180)*M_PI_2) {
			zone = 7;
			curlocx = 11;
			curlocy = 10;
			ultgoalx = 1;
			ultgoaly = 1;
		}
		else if (inittheta > (112.5/180)*M_PI_2 && inittheta < (157.5/180)*M_PI_2) {
			zone = 8;
			curlocx = 12;
			curlocy = 10;
			ultgoalx = 21;
			ultgoaly = 1;
		}
		else if (inittheta < (-22.5/180)*M_PI_2 && inittheta > (-67.5/180)*M_PI_2) {
			zone = 4;
			curlocx = 10;
			curlocy = 12;
			ultgoalx = 1;
			ultgoaly = 21;
		}
		else if (inittheta < (-67.5/180)*M_PI_2 && inittheta > (-112.5/180)*M_PI_2) {
			zone = 3;
			curlocx = 11;
			curlocy = 12;
			ultgoalx = 21;
			ultgoaly = 21;
		}
		else if (inittheta < (-112.5/180)*M_PI_2 && inittheta > (-157.5/180)*M_PI_2) {
			zone = 2;
```

```
                                        curlocx = 12;
                                        curlocy = 12;
                                        ultgoalx = 21;
                                        ultgoaly = 12;
                                }
                                else {
                                        zone = 1;
                                        curlocx = 12;
                                        curlocy = 11;
                                        ultgoalx = 21;
                                        ultgoaly = 2;
                                }

            // initialization has run
            init = true;
          } else {
            return;
          }

        }

        // If no collected or detected blocks set fingers
        // to open wide and raised position.
        if (!targetCollected && !targetDetected) {
          // set gripper
          std_msgs::Float32 angle;

          // open fingers
          angle.data = M_PI_2;

          fingerAnglePublish.publish(angle);
          angle.data = 0;

          // raise wrist
          wristAnglePublish.publish(angle);
        }

        // Select rotation or translation based on required adjustment
        switch(stateMachineState) {

        // If no adjustment needed, select new goal
        case STATE_MACHINE_TRANSFORM: {
          stateMachineMsg.data = "TRANSFORMING";

                        if (reset == 1) {
                                //code to go to two walls then go back to center
                                        wall = 1;
                                        if (zone == 1 || zone ==2) { //east
                                                goalLocation.theta = M_PI + atan2(currentLocation.y,
currentLocation.x);;
                                                goalLocation.x = 0.0;
```

```
                goalLocation.y = 0.0;
                reset = 0;
                stateMachineState = STATE_MACHINE_ROTATE;
                if (compround = 1) {
                        curlocx = 18;
                }
                else {
                        curlocx = 21;
                }
        }
        else if (zone == 3 || zone == 4) { //north
                goalLocation.theta = M_PI + atan2(currentLocation.y,
currentLocation.x);;
                goalLocation.x = 0.0;
                goalLocation.y = 0.0;
                reset = 0;
                stateMachineState = STATE_MACHINE_ROTATE;
                if (compround = 1) {
                        curlocy = 18;
                }
                else {
                        curlocy = 21;
                }
        }
        else if (zone == 5 || zone == 6) { //west
                goalLocation.theta = M_PI + atan2(currentLocation.y,
currentLocation.x);;
                goalLocation.x = 0.0;
                goalLocation.y = 0.0;
                stateMachineState = STATE_MACHINE_ROTATE;
                reset = 0;
                if (compround = 1) {
                        curlocx = 4;
                }
                else {
                        curlocx = 1;
                }
        }
        else { //south
                goalLocation.theta = M_PI + atan2(currentLocation.y,
currentLocation.x);;
                goalLocation.x = 0.0;
                goalLocation.y = 0.0;
                stateMachineState = STATE_MACHINE_ROTATE;
                reset = 0;
                if (compround = 1) {
                        curlocy = 4;
                }
                else {
                        curlocy = 1;
                }
```

```
                                        }

                        }
                // If returning with a target
                else if (targetCollected && !avoidingObstacle) {
                                // calculate the euclidean distance between
                                // centerLocation and currentLocation
                                dropOffController.setCenterDist(hypot(centerLocation.x - currentLocation.x,
centerLocation.y - currentLocation.y));
                                dropOffController.setDataLocations(centerLocation, currentLocation,
timerTimeElapsed);

                                DropOffResult result = dropOffController.getState();

                                if (result.timer) {
                                   timerStartTime = time(0);
                                   reachedCollectionPoint = true;
                                }

                                std_msgs::Float32 angle;

                                if (result.fingerAngle != -1) {
                                   angle.data = result.fingerAngle;
                                   fingerAnglePublish.publish(angle);
                                }

                                if (result.wristAngle != -1) {
                                   angle.data = result.wristAngle;
                                   wristAnglePublish.publish(angle);
                                }

                                if (result.reset) {
                                   timerStartTime = time(0);
                                   targetCollected = false;
                                   targetDetected = false;
                                   lockTarget = false;
                                   sendDriveCommand(0.0,0);

                                   // move back to transform step
                                   stateMachineState = STATE_MACHINE_TRANSFORM;
                                   reachedCollectionPoint = false;;
                                   centerLocationOdom = currentLocation;

                                   dropOffController.reset();
                                }
                                   else if (cubereturn == 1) {
                                                //goalLocation = result.centerGoal;
                                                if (zone == 1) {
                                                                if (curlocx == 12 && curlocy == 11) { //reached home space
and should see center
                                                                                goalLocation.theta = 2*M_PI_2;
```

```
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 11) { //one step up
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 12) { //two steps back
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 2;
                                        curlocy = curlocy - 2;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocy == 11) { //on neutral axis
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 13 || curlocx == 15 || curlocx == 17 ||
curlocx == 19 || curlocx == 21) { //clear to return
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;

                                }
                                else if ( curlocx == 14 || curlocx == 16 || curlocx == 18 ||
curlocx == 20) { //make one step back
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {//reset
                                        reset = 1;
                                }
                        }
                        if (zone == 2) {
                                if (curlocx == 12 && curlocy == 12) { //reached home space
and should see center
                                        goalLocation.theta = 2*M_PI_2;
```

```
                        goalLocation.x = currentLocation.x - 1;
                        goalLocation.y = currentLocation.y;
                        curlocx = curlocx - 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
                else if (curlocx == 11 && curlocy == 12) { //one step
                        goalLocation.theta = 3*M_PI_2;
                        goalLocation.x = currentLocation.x;
                        goalLocation.y = currentLocation.y - 1;
                        curlocy = curlocy - 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
                else if (curlocy == 12) { //on neutral axis
                        goalLocation.theta = 2*M_PI_2;
                        goalLocation.x = currentLocation.x - 1;
                        goalLocation.y = currentLocation.y;
                        curlocx = curlocx - 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
                else if (curlocx == 14 || curlocx == 16 || curlocx == 18 ||
curlocx == 20) { //clear to return
                        goalLocation.theta = 3*M_PI_2;
                        goalLocation.x = currentLocation.x;
                        goalLocation.y = currentLocation.y - 1;
                        curlocy = curlocy - 1;
                        stateMachineState = STATE_MACHINE_ROTATE;

                }
                else if ( curlocx == 15 || curlocx == 17 || curlocx == 19 ||
curlocx == 21) { //make one step back
                        goalLocation.theta = 2*M_PI_2;
                        goalLocation.x = currentLocation.x - 1;
                        goalLocation.y = currentLocation.y;
                        curlocx = curlocx - 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
                else {//reset
                        reset = 1;
                }

        }
        if (zone == 3) {
                if (curlocx == 11 && curlocy == 12) { //reached home space
and should see center
                        goalLocation.theta = 3*M_PI_2;
                        goalLocation.x = currentLocation.x;
                        goalLocation.y = currentLocation.y - 1;
                        curlocy = curlocy - 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
                else if (curlocx == 11 && curlocy == 11) { //one step up
```

```
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 10 && curlocy == 11) { //two steps back
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 2;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 2;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11) { //on neutral axis
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocy == 13 || curlocy == 15 || curlocy == 17 ||
curlocy == 19 || curlocy == 21) { //clear to return
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;

                                }
                                else if ( curlocy == 14 || curlocy == 16 || curlocy == 18 ||
curlocy == 20) { //make one step back
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {//reset
                                        reset = 1;
                                }
                        }
                        if (zone == 4) {
                                if (curlocx == 10 && curlocy == 12) { //reached home space
and should see center
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 12) { //one step up
```

```
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 10) { //on neutral axis
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocy == 13 || curlocy == 15 || curlocy == 17 ||
curlocy == 19 || curlocy == 21) { //clear to return
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;

                                }
                                else if ( curlocy == 14 || curlocy == 16 || curlocy == 18 ||
curlocy == 20) { //make one step back
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {//reset
                                        reset = 1;
                                }
                        }
                        if (zone == 5) {
                                if (curlocx == 10 && curlocy == 11) { //reached home space
and should see center
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 11) { //one step up
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 12) { //two steps back
```

```
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 2;
                                        curlocy = curlocy - 2;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocy == 11) { //on neutral axis
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 1 || curlocx == 3 || curlocx == 5 || curlocx
== 7 || curlocx == 9) { //clear to return
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;

                                }
                                else if ( curlocx == 2 || curlocx == 4 || curlocx == 6 || curlocx
== 8) { //make one step back
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {//reset
                                        reset = 1;
                                }
                        }
                        if (zone == 6) {
                                if (curlocx == 10 && curlocy == 10) { //reached home space
and should see center
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 11) { //one step up
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocy == 10) { //on neutral axis
```

```
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 2 || curlocx == 4 || curlocx == 6 || curlocx
== 8) { //clear to return
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;

                                }
                                else if ( curlocx == 1 || curlocx == 3 || curlocx == 5 || curlocx
== 7) { //make one step back
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {//reset
                                        reset = 1;
                                }
                        }
                        if (zone == 7) {
                                if (curlocx == 11 && curlocy == 10) { //reached home space
and should see center
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 11) { //one step up
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 12) { //two steps back
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x + 2;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 2;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11) { //on neutral axis
```

```
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocy == 9 || curlocy == 7 || curlocy == 5 || curlocy
== 3 || curlocy == 1) { //clear to return
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;

                                }
                                else if ( curlocy == 2 || curlocy == 4 || curlocy == 6 || curlocy
== 8) { //make one step back
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {//reset
                                        reset = 1;
                                }
                        }
                        if (zone == 8) {
                                if (curlocx == 12 && curlocy == 10) { //reached home space
and should see center
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 11 && curlocy == 10) { //one step up
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 12) { //on neutral axis
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
```

```
                                        else if (curlocy == 9 || curlocy == 7 || curlocy == 5 || curlocy
== 3 || curlocy == 1) { //clear to return
                                                goalLocation.theta = 3*M_PI_2;
                                                goalLocation.x = currentLocation.x - 1;
                                                goalLocation.y = currentLocation.y;
                                                curlocx = curlocx - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;

                                        }
                                        else if ( curlocy == 8 || curlocy == 6 || curlocy == 4 || curlocy
== 2) { //make one step back
                                                goalLocation.theta = 1*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y + 1;
                                                curlocy = curlocy + 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                        else {//reset
                                                reset = 1;
                                        }
                                }

                                stateMachineState = STATE_MACHINE_ROTATE;
                                timerStartTime = time(0);
                            }
                // we are in precision/timed driving
                            else {
                                goalLocation = currentLocation;
                                sendDriveCommand(result.cmdVel,result.angleError);
                                stateMachineState = STATE_MACHINE_TRANSFORM;

                                break;
                            }
                }
                //If angle between current and goal is significant
                //if error in heading is greater than 0.4 radians
                else if (fabs(angles::shortest_angular_distance(currentLocation.theta, goalLocation.theta)) >
rotateOnlyAngleTolerance) {
                    stateMachineState = STATE_MACHINE_ROTATE;
                }
                //If goal has not yet been reached drive and maintane heading
                else if (fabs(angles::shortest_angular_distance(currentLocation.theta, atan2(goalLocation.y -
currentLocation.y, goalLocation.x - currentLocation.x))) < M_PI_2) {
                    stateMachineState = STATE_MACHINE_SKID_STEER;
                }
                //Otherwise, drop off target and select new random uniform heading
                //If no targets have been detected, assign a new goal

//EDIT                  ASSIGN NEW GOAL
                else if (!targetDetected && timerTimeElapsed > returnToSearchDelay) {
                    //USE COORDINATE SYSTEM LIKE CHESSBOARD WITH DISCRETE 1mx1m spaces.
```

```
		if (newspace == 0) { //checks to see if this is first time to space. 0 signifies never reached
before.

			if (turn < 4) {
				//perform a 360 scan of the area in 90deg increments
				goalLocation.theta = currentLocation.theta + M_PI_2;
				turn = turn + 1;
				stateMachineState = STATE_MACHINE_ROTATE;
			}
			else {
				turn = 0;
				newspace = 1;
				//proceed to next step of assigning new goal location
			}

		}

		//resets rover from endpoint of one zone to the starting point of the next zone
*****ADD IN CURLOC CHANGE AFTER EACH COMMAND*******
		else if (zonereset == 1) {
			// ZONE 8 to 1
			if (zone == 1) {
				if (resetstep == 1) {
					if (curlocx != 12){
						goalLocation.theta = 2*M_PI_2;
						goalLocation.x = currentLocation.x - 1;
						goalLocation.y = currentLocation.y;
						stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
					}
					else {
						resetstep = 2;
					stateMachineState = STATE_MACHINE_TRANSFORM;
					}
				}
				else if (resetstep == 2) {
					if (curlocy != 10){
						goalLocation.theta = 1*M_PI_2;
						goalLocation.x = currentLocation.x;
						goalLocation.y = currentLocation.y + 1;
						stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
					}
					else {
						resetstep = 3;
					stateMachineState = STATE_MACHINE_TRANSFORM;
					}
				}
				else if (resetstep == 3) { //make one step from space 8 to space 1
					goalLocation.theta = 1*M_PI_2;
```

```
                        goalLocation.x = currentLocation.x;
                        goalLocation.y = currentLocation.y + 1;
                        zonereset = 0;
                        resetstep = 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
        }
        // ZONE 1 to 2
        else if (zone == 2) {
                if (resetstep == 1) {
                        if (curlocy != 11){
                                goalLocation.theta = 1*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y + 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                        }
                        else {
                                resetstep = 2;
                        stateMachineState = STATE_MACHINE_TRANSFORM;
                        }
                }
                else if (resetstep == 2) {
                        if (curlocx != 12){
                                goalLocation.theta = 2*M_PI_2;
                                goalLocation.x = currentLocation.x - 1;
                                goalLocation.y = currentLocation.y;
                                stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                        }
                        else {
                                resetstep = 3;
                        stateMachineState = STATE_MACHINE_TRANSFORM;
                        }
                }
                else if (resetstep == 3) { //make one step from space 8 to space 1
                        goalLocation.theta = 1*M_PI_2;
                        goalLocation.x = currentLocation.x;
                        goalLocation.y = currentLocation.y + 1;
                        zonereset = 0;
                        resetstep = 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
        }
        // ZONE 2 to 3
        else if (zone == 3) {
                if (resetstep == 1) {
                        if (curlocy != 12){
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
```

```
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 2;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 2) {
                                if (curlocx != 12){
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 3;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 3) { //make one step from space 8 to space 1
                                goalLocation.theta = 2*M_PI_2;
                                goalLocation.x = currentLocation.x - 1;
                                goalLocation.y = currentLocation.y;
                                zonereset = 0;
                                resetstep = 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }
                // ZONE 3 to 4
                else if (zone == 1) {
                        if (resetstep == 1) {
                                if (curlocx != 11){
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 2;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 2) {
                                if (curlocy != 12){
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
```

```
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 3;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 3) { //make one step from space 8 to space 1
                                goalLocation.theta = 2*M_PI_2;
                                goalLocation.x = currentLocation.x - 1;
                                goalLocation.y = currentLocation.y;
                                zonereset = 0;
                                resetstep = 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }
                // ZONE 4 to 5
                else if (zone == 1) {
                        if (resetstep == 1) {
                                if (curlocx != 10){
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 2;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 2) {
                                if (curlocy != 12){
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 3;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 3) { //make one step from space 8 to space 1
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
                                zonereset = 0;
                                resetstep = 1;
```

```
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
        }
        // ZONE 5 to 6
        else if (zone == 1) {
                if (resetstep == 1) {
                        if (curlocy != 11){
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                        }
                        else {
                                resetstep = 2;
                        stateMachineState = STATE_MACHINE_TRANSFORM;
                        }
                }
                else if (resetstep == 2) {
                        if (curlocx != 10){
                                goalLocation.theta = 0*M_PI_2;
                                goalLocation.x = currentLocation.x + 1;
                                goalLocation.y = currentLocation.y;
                                stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                        }
                        else {
                                resetstep = 3;
                        stateMachineState = STATE_MACHINE_TRANSFORM;
                        }
                }
                else if (resetstep == 3) { //make one step from space 8 to space 1
                        goalLocation.theta = 3*M_PI_2;
                        goalLocation.x = currentLocation.x;
                        goalLocation.y = currentLocation.y - 1;
                        zonereset = 0;
                        resetstep = 1;
                        stateMachineState = STATE_MACHINE_ROTATE;
                }
        }
        // ZONE 6 to 7
        else if (zone == 1) {
                if (resetstep == 1) {
                        if (curlocy != 10){
                                goalLocation.theta = 2*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y + 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                        }
                        else {
```

```
                                        resetstep = 2;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 2) {
                                if (curlocx != 10){
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 3;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 3) { //make one step from space 8 to space 1
                                goalLocation.theta = 0*M_PI_2;
                                goalLocation.x = currentLocation.x + 1;
                                goalLocation.y = currentLocation.y;
                                zonereset = 0;
                                resetstep = 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }
                // ZONE 7 to 8
                else if (zone == 1) {
                        if (resetstep == 1) {
                                if (curlocx != 11){
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 2;
                                stateMachineState = STATE_MACHINE_TRANSFORM;
                                }
                        }
                        else if (resetstep == 2) {
                                if (curlocy != 10){
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
//move to rotate stage
                                }
                                else {
                                        resetstep = 3;
```

```
                                        stateMachineState = STATE_MACHINE_TRANSFORM;
                                        }
                                }
                                else if (resetstep == 3) { //make one step from space 8 to space 1
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        zonereset = 0;
                                        resetstep = 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                        }

                }

                //have 2 variables intermediate goal for next step and final goal for end of path.
                else if (ultgoalx == curlocx && ultgoaly == curlocy) { //assign new ult goal as end of
desired path has been reached
                        if (zone == 1) {
                                zone = 2;
                                zonereset = 1;
                        }
                        else if (zone == 2) {
                                zone = 3;
                                zonereset = 1;
                        }
                        else if (zone == 3) {
                                zone = 4;
                                zonereset = 1;
                        }
                        else if (zone == 4) {
                                zone = 5;
                                zonereset = 1;
                        }
                        else if (zone == 5) {
                                zone = 6;
                                zonereset = 1;
                        }
                        else if (zone == 6) {
                                zone = 7;
                                zonereset = 1;
                        }
                        else if (zone == 7) {
                                zone = 8;
                                zonereset = 1;
                        }
                        else if (zone == 8) {
                                zone = 1;
                                zonereset = 1;
                        }
```

```
                stateMachineState = STATE_MACHINE_TRANSFORM; //reset transform state to get into
zonereset loop
                }
                else if (lastcube == 1) {
                        if (zone == 1 || zone == 2) {
                                if (intgoalx != curlocx) {
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {
                                        if (zone == 2) {
                                                if (intgoaly != curlocy) {
                                                        goalLocation.theta = 1*M_PI_2;
                                                        goalLocation.x = currentLocation.x;
                                                        goalLocation.y = currentLocation.y + 1;
                                                        curlocy = curlocy + 1;
                                                        stateMachineState =
STATE_MACHINE_ROTATE;
                                                }
                                                else {
                                                        lastcube = 0;
                                                        stateMachineState =
STATE_MACHINE_TRANSFORM;
                                                }
                                        }
                                        else {
                                                if (intgoaly != curlocy) {
                                                        goalLocation.theta = 3*M_PI_2;
                                                        goalLocation.x = currentLocation.x;
                                                        goalLocation.y = currentLocation.y - 1;
                                                        curlocy = curlocy - 1;
                                                        stateMachineState =
STATE_MACHINE_ROTATE;
                                                }
                                                else {
                                                        lastcube = 0;
                                                        stateMachineState =
STATE_MACHINE_TRANSFORM;
                                                }
                                        }
                                }
                        }
                        if (zone == 3 || zone == 4) {
                                if (intgoaly != curlocy) {
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
```

```
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {
                                        if (zone == 3) {
                                                if (intgoalx != curlocx) {
                                                        goalLocation.theta = 0*M_PI_2;
                                                        goalLocation.x = currentLocation.x + 1;
                                                        goalLocation.y = currentLocation.y;
                                                        curlocx = curlocx + 1;
                                                        stateMachineState =
STATE_MACHINE_ROTATE;
                                                }
                                                else {
                                                        lastcube = 0;
                                                        stateMachineState =
STATE_MACHINE_TRANSFORM;
                                                }
                                        }
                                        else {
                                                if (intgoalx != curlocx) {
                                                        goalLocation.theta = 2*M_PI_2;
                                                        goalLocation.x = currentLocation.x - 1;
                                                        goalLocation.y = currentLocation.y;
                                                        curlocx = curlocx - 1;
                                                        stateMachineState =
STATE_MACHINE_ROTATE;
                                                }
                                                else {
                                                        lastcube = 0;
                                                        stateMachineState =
STATE_MACHINE_TRANSFORM;
                                                }
                                        }
                                }
                        }
                        if (zone == 5 || zone == 6) {
                                if (intgoalx != curlocx) {
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else {
                                        if (zone == 5) {
                                                if (intgoaly != curlocy) {
                                                        goalLocation.theta = 1*M_PI_2;
                                                        goalLocation.x = currentLocation.x;
                                                        goalLocation.y = currentLocation.y + 1;
                                                        curlocy = curlocy + 1;
```

```
                                                stateMachineState =
STATE_MACHINE_ROTATE;
                                        }
                                        else {
                                                lastcube = 0;
                                                stateMachineState =
STATE_MACHINE_TRANSFORM;
                                        }
                                }
                                else {
                                        if (intgoaly != curlocy) {
                                                goalLocation.theta = 3*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y - 1;
                                                curlocy = curlocy - 1;
                                                stateMachineState =
STATE_MACHINE_ROTATE;
                                        }
                                        else {
                                                lastcube = 0;
                                                stateMachineState =
STATE_MACHINE_TRANSFORM;
                                        }
                                }
                        }
                }
                else {
                        if (intgoaly != curlocy) {
                                goalLocation.theta = 0*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
                                curlocy = curlocy - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                        else {
                                if (zone == 8) {
                                        if (intgoalx != curlocx) {
                                                goalLocation.theta = 0*M_PI_2;
                                                goalLocation.x = currentLocation.x + 1;
                                                goalLocation.y = currentLocation.y;
                                                curlocx = curlocx + 1;
                                                stateMachineState =
STATE_MACHINE_ROTATE;
                                        }
                                        else {
                                                lastcube = 0;
                                                stateMachineState =
STATE_MACHINE_TRANSFORM;
                                        }
                                }
                                else {
```

```
                                                if (intgoalx != curlocx) {
                                                        goalLocation.theta = 2*M_PI_2;
                                                        goalLocation.x = currentLocation.x - 1;
                                                        goalLocation.y = currentLocation.y;
                                                        curlocx = curlocx - 1;
                                                        stateMachineState =
STATE_MACHINE_ROTATE;
                                                }
                                                else {
                                                        lastcube = 0;
                                                        stateMachineState =
STATE_MACHINE_TRANSFORM;
                                                }
                                        }
                                }
                        }
                }
                else { //assign the next step for int goal to get to ult goal
                        newspace = 0;
                        //Path by zone
                        if (zone == 1) {
                                if (compround == 1) {
                                        ultgoalx = 18;
                                        ultgoaly = 11;
                                }
                                if (curlocy != 11 && (curlocx == 14 || curlocx == 16 || curlocx == 18 ||
curlocx == 20)) { //go north
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else if (curlocx == 13 || curlocx == 15 || curlocx == 17 || curlocx == 19
|| curlocx == 21) {
                                        if ((curlocx == 13 && curlocy == 10) || (curlocx == 15 &&
curlocy == 8) || (curlocx == 17 && curlocy == 6) || (curlocx == 19 && curlocy == 4)) { //go east
                                                goalLocation.theta = 0;
                                                goalLocation.x = currentLocation.x + 1;
                                                goalLocation.y = currentLocation.y;
                                                curlocx = curlocx + 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                        else { //go south
                                                goalLocation.theta = 3*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y - 1;
                                                curlocy = curlocy - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                }
```

```
                        else { //go east
                                goalLocation.theta = 0;
                                goalLocation.x = currentLocation.x + 1;
                                goalLocation.y = currentLocation.y;
                                curlocx = curlocx + 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }

                else if (zone == 2) {
                        if (compround == 1) {
                                ultgoalx = 18;
                                ultgoaly = 18;
                        }
                        //first is direction to neutral axis
                        if (curlocy != 12 && (curlocx == 15 || curlocx == 17 || curlocx == 19 ||
curlocx == 21)) { //go south
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
                                curlocy = curlocy - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                        //second is opposite of first
                        else if (curlocx == 14 || curlocx == 16 || curlocx == 18 || curlocx ==
20) {
                                if ((curlocx == 14 && curlocy == 14) || (curlocx == 16 &&
curlocy == 16) || (curlocx == 18 && curlocy == 18) || (curlocx == 20 && curlocy == 20)) { //go east
                                        goalLocation.theta = 0;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else { //go north
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                        }
                        //third is remaining direction
                        else { //go east
                                goalLocation.theta = 0;
                                goalLocation.x = currentLocation.x + 1;
                                goalLocation.y = currentLocation.y;
                                curlocx = curlocx + 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }
```

```
                else if (zone == 3) {
                        if (compround == 1) {
                                ultgoalx = 11;
                                ultgoaly = 18;
                        }
                        //first is direction to neutral axis
                        if (curlocx != 11 && (curlocy == 14 || curlocy == 16 || curlocy == 18 ||
curlocy == 20)) { //go west
                                goalLocation.theta = 2*M_PI_2;
                                goalLocation.x = currentLocation.x - 1;
                                goalLocation.y = currentLocation.y;
                                curlocx = curlocx - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                        //second is opposite of first
                        else if (curlocy == 13 || curlocy == 15 || curlocy == 17 || curlocy == 19
|| curlocy == 21) {
                                if ((curlocx == 13 && curlocy == 13) || (curlocx == 15 &&
curlocy == 15) || (curlocx == 17 && curlocy == 17) || (curlocx == 19 && curlocy == 19)) { //go north
                                        goalLocation.theta = 1*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y + 1;
                                        curlocy = curlocy + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else { // go east
                                        goalLocation.theta = 0;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                        }
                        //third is remaining direction
                        else { //go north
                                goalLocation.theta = 1*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y + 1;
                                curlocy = curlocy + 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }

                else if (zone == 4) {
                        if (compround == 1) {
                                ultgoalx = 10;
                                ultgoaly = 18;
                        }
                        //first is direction to neutral axis
```

```
                                        if (curlocx != 10 && (curlocy == 14 || curlocy == 16 || curlocy == 18 ||
curlocy == 20)) { //go east
                                                goalLocation.theta = 0*M_PI_2;
                                                goalLocation.x = currentLocation.x + 1;
                                                goalLocation.y = currentLocation.y;
                                                curlocx = curlocx + 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                        //second is opposite of first
                                        else if (curlocy == 13 || curlocy == 15 || curlocy == 17 || curlocy == 19
|| curlocy == 21) {
                                                if ((curlocx == 8 && curlocy == 13) || (curlocx == 6 && curlocy
== 15) || (curlocx == 4 && curlocy == 17) || (curlocx == 2 && curlocy == 19)) { //go north
                                                        goalLocation.theta = 1*M_PI_2;
                                                        goalLocation.x = currentLocation.x;
                                                        goalLocation.y = currentLocation.y + 1;
                                                        curlocy = curlocy + 1;
                                                        stateMachineState = STATE_MACHINE_ROTATE;
                                                }
                                                else { //go west
                                                        goalLocation.theta = 2*M_PI_2;
                                                        goalLocation.x = currentLocation.x - 1;
                                                        goalLocation.y = currentLocation.y;
                                                        curlocx = curlocx - 1;
                                                        stateMachineState = STATE_MACHINE_ROTATE;
                                                }
                                        }
                                        //third is remaining direction
                                        else { //go north
                                                goalLocation.theta = 1*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y + 1;
                                                curlocy = curlocy + 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                }

                                else if (zone == 5) { //
                                        if (compround == 1) {
                                                ultgoalx = 4;
                                                ultgoaly = 11;
                                        }
                                        //first is direction to neutral axis
                                        if (curlocy != 11 && (curlocx == 8 || curlocx == 6 || curlocx == 4 ||
curlocx == 2)) { //go south
                                                goalLocation.theta = 3*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y - 1;
                                                curlocy = curlocy - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
```

```
                                        //second is opposite of first
                                        else if (curlocx == 9 || curlocx == 7 || curlocx == 5 || curlocx == 3 ||
curlocx == 1) {
                                                if ((curlocx == 9 && curlocy == 12) || (curlocx == 7 && curlocy
== 14) || (curlocx == 5 && curlocy == 16) || (curlocx == 3 && curlocy == 18)) { //go west
                                                        goalLocation.theta = 2*M_PI_2;
                                                        goalLocation.x = currentLocation.x - 1;
                                                        goalLocation.y = currentLocation.y;
                                                        curlocx = curlocx - 1;
                                                        stateMachineState = STATE_MACHINE_ROTATE;
                                                }
                                                else { //go north
                                                        goalLocation.theta = 1*M_PI_2;
                                                        goalLocation.x = currentLocation.x;
                                                        goalLocation.y = currentLocation.y + 1;
                                                        curlocy = curlocy + 1;
                                                        stateMachineState = STATE_MACHINE_ROTATE;
                                                }
                                        }
                                        //third is remaining direction
                                        else { //go west
                                                goalLocation.theta = 2*M_PI_2;
                                                goalLocation.x = currentLocation.x - 1;
                                                goalLocation.y = currentLocation.y;
                                                curlocx = curlocx - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                }
                                else if (zone == 6) { //
                                        if (compround == 1) {
                                                ultgoalx = 4;
                                                ultgoaly = 4;
                                        }
                                        //first is direction to neutral axis
                                        if (curlocy != 10 && (curlocx == 7 || curlocx == 5 || curlocx == 3 ||
curlocx == 1)) { //go north
                                                goalLocation.theta = 1*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y + 1;
                                                curlocy = curlocy + 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                        //second is opposite of first
                                        else if (curlocx == 8 || curlocx == 6 || curlocx == 4 || curlocx == 2) {
                                                if ((curlocx == 8 && curlocy == 8) || (curlocx == 6 && curlocy
== 6) || (curlocx == 4 && curlocy == 4) || (curlocx ==2 && curlocy == 2)) { //go west
                                                        goalLocation.theta = 2*M_PI_2;
                                                        goalLocation.x = currentLocation.x - 1;
                                                        goalLocation.y = currentLocation.y;
                                                        curlocx = curlocx - 1;
                                                        stateMachineState = STATE_MACHINE_ROTATE;
```

```
                                        }
                                        else { //go south
                                                goalLocation.theta = 3*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y - 1;
                                                curlocy = curlocy - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                }
                                //third is remaining direction
                                else { //go west
                                        goalLocation.theta = 2*M_PI_2;
                                        goalLocation.x = currentLocation.x - 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                        }

                        else if (zone == 7) {
                                if (compround == 1) {
                                        ultgoalx = 11;
                                        ultgoaly = 4;
                                }
                                //first is direction to neutral axis
                                if (curlocx != 11 && (curlocy == 8 || curlocy == 6 || curlocy == 4 ||
curlocy == 2)) { //go east
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                //second is opposite of first
                                else if (curlocy == 9 || curlocy == 7 || curlocy == 5 || curlocy == 3 ||
curlocy == 1) {
                                        if ((curlocx == 9 && curlocy == 9) || (curlocx == 7 && curlocy
== 7) || (curlocx == 5 && curlocy == 5) || (curlocx == 3 && curlocy == 3)) { //go south
                                                goalLocation.theta = 3*M_PI_2;
                                                goalLocation.x = currentLocation.x;
                                                goalLocation.y = currentLocation.y - 1;
                                                curlocy = curlocy - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
                                        else { //go west
                                                goalLocation.theta = 2*M_PI_2;
                                                goalLocation.x = currentLocation.x - 1;
                                                goalLocation.y = currentLocation.y;
                                                curlocx = curlocx - 1;
                                                stateMachineState = STATE_MACHINE_ROTATE;
                                        }
```

```
                        }
                        //third is remaining direction
                        else { //go south
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
                                curlocy = curlocy - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                }

                else if (zone == 8) {
                        if (compround == 1) {
                                ultgoalx = 12;
                                ultgoaly = 4;
                        }
                        //first is direction to neutral axis
                        if (curlocx != 12 && (curlocy == 8 || curlocy == 6 || curlocy == 4 ||
curlocy == 2)) { //go west
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x - 1;
                                goalLocation.y = currentLocation.y;
                                curlocx = curlocx - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
                        }
                        //second is opposite of first
                        else if (curlocy == 9 || curlocy == 7 || curlocy == 5 || curlocy == 3 ||
curlocy == 1) {
                                if ((curlocx == 14 && curlocy == 9) || (curlocx == 16 &&
curlocy == 7) || (curlocx == 18 && curlocy == 5) || (curlocx == 20 && curlocy == 3)) { //go south
                                        goalLocation.theta = 3*M_PI_2;
                                        goalLocation.x = currentLocation.x;
                                        goalLocation.y = currentLocation.y - 1;
                                        curlocy = curlocy - 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                                else { //go east
                                        goalLocation.theta = 0*M_PI_2;
                                        goalLocation.x = currentLocation.x + 1;
                                        goalLocation.y = currentLocation.y;
                                        curlocx = curlocx + 1;
                                        stateMachineState = STATE_MACHINE_ROTATE;
                                }
                        }
                        //third is remaining direction
                        else { //go south
                                goalLocation.theta = 3*M_PI_2;
                                goalLocation.x = currentLocation.x;
                                goalLocation.y = currentLocation.y - 1;
                                curlocy = curlocy - 1;
                                stateMachineState = STATE_MACHINE_ROTATE;
```

```
                    }
                }

            }


            //goalLocation.theta = 0;

            //goalLocation.theta = 1*M_PI_2;

            //goalLocation.theta = 2*M_PI_2;

            //goalLocation.theta = 3*M_PI_2;
        }

        //Purposefully fall through to next case without breaking
    }

    // Calculate angle between currentLocation.theta and goalLocation.theta
    // Rotate left or right depending on sign of angle
    // Stay in this state until angle is minimized
    case STATE_MACHINE_ROTATE: {
        stateMachineMsg.data = "ROTATING";
        // Calculate the diffrence between current and desired
        // heading in radians.
        float errorYaw = angles::shortest_angular_distance(currentLocation.theta, goalLocation.theta);

        // If angle > 0.4 radians rotate but dont drive forward.
        if (fabs(angles::shortest_angular_distance(currentLocation.theta, goalLocation.theta)) >
rotateOnlyAngleTolerance) {
            // rotate but dont drive  0.05 is to prevent turning in reverse
            sendDriveCommand(0.01, errorYaw/2);
            break;
        } else {
            // move to differential drive step
            sendDriveCommand(0.0, 0.0);
            stateMachineState = STATE_MACHINE_ROTATE;
            //fall through on purpose.
        }
    }

    // Calculate angle between currentLocation.x/y and goalLocation.x/y
    // Drive forward
    // Stay in this state until angle is at least PI/2
    case STATE_MACHINE_SKID_STEER: {
        stateMachineMsg.data = "SKID_STEER";

        // calculate the distance between current and desired heading in radians
        float errorYaw = angles::shortest_angular_distance(currentLocation.theta, goalLocation.theta);

        // goal not yet reached drive while maintaining proper heading.
```

```
        if (fabs(angles::shortest_angular_distance(currentLocation.theta, atan2(goalLocation.y -
currentLocation.y, goalLocation.x - currentLocation.x))) < M_PI_2) {
            // drive and turn simultaniously
            sendDriveCommand(searchVelocity, errorYaw/4);
        }
        // goal is reached but desired heading is still wrong turn only
        else if (fabs(angles::shortest_angular_distance(currentLocation.theta, goalLocation.theta)) > 0.1) {
            // rotate but dont drive
            sendDriveCommand(0.0, errorYaw/2);
        }
        else {
            // stop
            sendDriveCommand(0.0, 0.0);
            avoidingObstacle = false;
            // move back to transform step
            stateMachineState = STATE_MACHINE_TRANSFORM;
        }

        break;
    }

    case STATE_MACHINE_PICKUP: {
        stateMachineMsg.data = "PICKUP";

        PickUpResult result;

        // we see a block and have not picked one up yet
        if (targetDetected && !targetCollected) {
            result = pickUpController.pickUpSelectedTarget(blockBlock);
            sendDriveCommand(result.cmdVel,result.angleError);
            std_msgs::Float32 angle;

            if (result.fingerAngle != -1) {
                angle.data = result.fingerAngle;
                fingerAnglePublish.publish(angle);
            }

            if (result.wristAngle != -1) {
                angle.data = result.wristAngle;

                // raise wrist
                wristAnglePublish.publish(angle);
            }

            if (result.giveUp) {
                targetDetected = false;
                stateMachineState = STATE_MACHINE_TRANSFORM;
                sendDriveCommand(0,0);
                pickUpController.reset();
            }
```

```
        if (result.pickedUp) {
          pickUpController.reset();

          // assume target has been picked up by gripper
          targetCollected = true;
          result.pickedUp = false;
          stateMachineState = STATE_MACHINE_TRANSFORM;

                              intgoalx = curlocx;
                              intgoaly = curlocy;
                              lastcube = 1;
                              cubereturn = 1;

          // lower wrist to avoid ultrasound sensors
          std_msgs::Float32 angle;
          angle.data = 0.8;
          wristAnglePublish.publish(angle);
          sendDriveCommand(0.0,0);

          return;
        }
      } else {
        stateMachineState = STATE_MACHINE_TRANSFORM;
      }

      break;
    }

    case STATE_MACHINE_DROPOFF: {
      stateMachineMsg.data = "DROPOFF";
                    cubereturn = 0;
      break;
    }

    default: {
      break;
    }

    } /* end of switch() */
}
// mode is NOT auto
else {
    // publish current state for the operator to see
    stateMachineMsg.data = "WAITING";
}

// publish state machine string for user, only if it has changed, though
if (strcmp(stateMachineMsg.data.c_str(), prev_state_machine) != 0) {
    stateMachinePublish.publish(stateMachineMsg);
    sprintf(prev_state_machine, "%s", stateMachineMsg.data.c_str());
}
```

```
}

void sendDriveCommand(double linearVel, double angularError)
{
   velocity.linear.x = linearVel,
   velocity.angular.z = angularError;

   // publish the drive commands
   driveControlPublish.publish(velocity);
}

/************************
 * ROS CALLBACK HANDLERS *
 ************************/

void targetHandler(const apriltags_ros::AprilTagDetectionArray::ConstPtr& message) {

   // If in manual mode do not try to automatically pick up the target
   if (currentMode == 1 || currentMode == 0) return;

   // if a target is detected and we are looking for center tags
   if (message->detections.size() > 0 && !reachedCollectionPoint) {
      float cameraOffsetCorrection = 0.020; //meters;

      centerSeen = false;
      double count = 0;
      double countRight = 0;
      double countLeft = 0;

      // this loop is to get the number of center tags
      for (int i = 0; i < message->detections.size(); i++) {
         if (message->detections[i].id == 256) {
            geometry_msgs::PoseStamped cenPose = message->detections[i].pose;

            // checks if tag is on the right or left side of the image
            if (cenPose.pose.position.x + cameraOffsetCorrection > 0) {
               countRight++;

            } else {
               countLeft++;
            }

            centerSeen = true;
            count++;
         }
      }

      if (centerSeen && targetCollected) {
         stateMachineState = STATE_MACHINE_TRANSFORM;
         goalLocation = currentLocation;
      }
```

```
dropOffController.setDataTargets(count,countLeft,countRight);

// if we see the center and we dont have a target collected
if (centerSeen && !targetCollected) {

   float centeringTurn = 0.15; //radians
   stateMachineState = STATE_MACHINE_ROTATE;

                goalLocation.theta = currentLocation.theta + 2*M_PI_2;
                goalLocation.x = currentLocation.x;
                goalLocation.y = currentLocation.y;

                        float inittheta = angles::shortest_angular_distance(currentLocation.theta, 0);

                        if (abs(inittheta) < (22.5/180)*M_PI_2) {
                                curlocx = 10;
                                curlocy = 11;
                        }
                        else if (inittheta > (22.5/180)*M_PI_2 && inittheta < (67.5/180)*M_PI_2) {
                                curlocx = 10;
                                curlocy = 10;
                        }
                        else if (inittheta > (67.5/180)*M_PI_2 && inittheta < (112.5/180)*M_PI_2) {
                                curlocx = 11;
                                curlocy = 10;
                        }
                        else if (inittheta > (112.5/180)*M_PI_2 && inittheta < (157.5/180)*M_PI_2) {
                                curlocx = 12;
                                curlocy = 10;
                        }
                        else if (inittheta < (-22.5/180)*M_PI_2 && inittheta > (-67.5/180)*M_PI_2) {
                                curlocx = 10;
                                curlocy = 12;
                        }
                        else if (inittheta > (-67.5/180)*M_PI_2 && inittheta > (-112.5/180)*M_PI_2) {
                                curlocx = 11;
                                curlocy = 12;
                        }
                        else if (inittheta < (-112.5/180)*M_PI_2 && inittheta > (-157.5/180)*M_PI_2) {
                                curlocx = 12;
                                curlocy = 12;
                        }
                        else {
                                curlocx = 12;
                                curlocy = 11;
                        }


   // this code keeps the robot from driving over
```

```
          // the center when searching for blocks
//          if (right) {
 //            // turn away from the center to the left if just driving
  //           // around/searching.
   //          goalLocation.theta += centeringTurn;
    //       } else {
     //           // turn away from the center to the right if just driving
     //           // around/searching.
      //       goalLocation.theta -= centeringTurn;
       //     //}

          // continues an interrupted search
       //   goalLocation = searchController.continueInterruptedSearch(currentLocation, goalLocation);

          targetDetected = false;
          pickUpController.reset();

          return;
        }
    }
    // end found target and looking for center tags

    // found a target april tag and looking for april cubes;
    // with safety timer at greater than 5 seconds.
    PickUpResult result;

    if (message->detections.size() > 0 && !targetCollected && timerTimeElapsed > 5) {
        targetDetected = true;

        // pickup state so target handler can take over driving.
        stateMachineState = STATE_MACHINE_PICKUP;
        result = pickUpController.selectTarget(message);

        std_msgs::Float32 angle;

        if (result.fingerAngle != -1) {
            angle.data = result.fingerAngle;
            fingerAnglePublish.publish(angle);
        }

        if (result.wristAngle != -1) {
            angle.data = result.wristAngle;
            wristAnglePublish.publish(angle);
        }
    }
}

void modeHandler(const std_msgs::UInt8::ConstPtr& message) {
    currentMode = message->data;
    sendDriveCommand(0.0, 0.0);
}
```

```
void obstacleHandler(const std_msgs::UInt8::ConstPtr& message) {
  if ((!targetDetected || targetCollected) && (message->data > 0)) {
    // obstacle on right side
    if (message->data == 1) {
      // select new heading 0.2 radians to the left
      goalLocation.theta = currentLocation.theta + 0.6;
    }

    // obstacle in front or on left side
    else if (message->data == 2) {
      // select new heading 0.2 radians to the right
      goalLocation.theta = currentLocation.theta + 0.6;
    }

    // continues an interrupted search
    goalLocation = searchController.continueInterruptedSearch(currentLocation, goalLocation);

    // switch to transform state to trigger collision avoidance
    stateMachineState = STATE_MACHINE_ROTATE;

    avoidingObstacle = true;
  }

  // the front ultrasond is blocked very closely. 0.14m currently
  if (message->data == 4) {
    blockBlock = true;
  } else {
    blockBlock = false;
  }
}

void odometryHandler(const nav_msgs::Odometry::ConstPtr& message) {
  //Get (x,y) location directly from pose
  currentLocation.x = message->pose.pose.position.x;
  currentLocation.y = message->pose.pose.position.y;

  //Get theta rotation by converting quaternion orientation to pitch/roll/yaw
  tf::Quaternion q(message->pose.pose.orientation.x, message->pose.pose.orientation.y, message-
>pose.pose.orientation.z, message->pose.pose.orientation.w);
  tf::Matrix3x3 m(q);
  double roll, pitch, yaw;
  m.getRPY(roll, pitch, yaw);
  currentLocation.theta = yaw;
}

void mapHandler(const nav_msgs::Odometry::ConstPtr& message) {
  //Get (x,y) location directly from pose
  currentLocationMap.x = message->pose.pose.position.x;
  currentLocationMap.y = message->pose.pose.position.y;
```

```
   //Get theta rotation by converting quaternion orientation to pitch/roll/yaw
   tf::Quaternion q(message->pose.pose.orientation.x, message->pose.pose.orientation.y, message-
>pose.pose.orientation.z, message->pose.pose.orientation.w);
   tf::Matrix3x3 m(q);
   double roll, pitch, yaw;
   m.getRPY(roll, pitch, yaw);
   currentLocationMap.theta = yaw;
}

void joyCmdHandler(const sensor_msgs::Joy::ConstPtr& message) {
   if (currentMode == 0 || currentMode == 1) {
      sendDriveCommand(abs(message->axes[4]) >= 0.1 ? message->axes[4] : 0, abs(message->axes[3]) >=
0.1 ? message->axes[3] : 0);
   }
}


void publishStatusTimerEventHandler(const ros::TimerEvent&) {
   std_msgs::String msg;
   msg.data = "online";
   status_publisher.publish(msg);
}


void targetDetectedReset(const ros::TimerEvent& event) {
   targetDetected = false;

   std_msgs::Float32 angle;
   angle.data = 0;

   // close fingers
   fingerAnglePublish.publish(angle);

   // raise wrist
   wristAnglePublish.publish(angle);
}

void sigintEventHandler(int sig) {
   // All the default sigint handler does is call shutdown()
   ros::shutdown();
}

void mapAverage() {
   // store currentLocation in the averaging array
   mapLocation[mapCount] = currentLocationMap;
   mapCount++;

   if (mapCount >= mapHistorySize) {
      mapCount = 0;
   }
```

```
  double x = 0;
  double y = 0;
  double theta = 0;

  // add up all the positions in the array
  for (int i = 0; i < mapHistorySize; i++) {
    x += mapLocation[i].x;
    y += mapLocation[i].y;
    theta += mapLocation[i].theta;
  }

  // find the average
  x = x/mapHistorySize;
  y = y/mapHistorySize;

  // Get theta rotation by converting quaternion orientation to pitch/roll/yaw
  theta = theta/100;
  currentLocationAverage.x = x;
  currentLocationAverage.y = y;
  currentLocationAverage.theta = theta;


  // only run below code if a centerLocation has been set by initilization
  if (init) {
    // map frame
    geometry_msgs::PoseStamped mapPose;

    // setup msg to represent the center location in map frame
    mapPose.header.stamp = ros::Time::now();

    mapPose.header.frame_id = publishedName + "/map";
    mapPose.pose.orientation = tf::createQuaternionMsgFromRollPitchYaw(0, 0,
centerLocationMap.theta);
    mapPose.pose.position.x = centerLocationMap.x;
    mapPose.pose.position.y = centerLocationMap.y;
    geometry_msgs::PoseStamped odomPose;
    string x = "";

    try { //attempt to get the transform of the center point in map frame to odom frame.
      tfListener->waitForTransform(publishedName + "/map", publishedName + "/odom",
ros::Time::now(), ros::Duration(1.0));
      tfListener->transformPose(publishedName + "/odom", mapPose, odomPose);
    }

    catch(tf::TransformException& ex) {
      ROS_INFO("Received an exception trying to transform a point from \"map\" to \"odom\": %s",
ex.what());
      x = "Exception thrown " + (string)ex.what();
      std_msgs::String msg;
      stringstream ss;
      ss << "Exception in mapAverage() " + (string)ex.what();
```

```
            msg.data = ss.str();
            infoLogPublisher.publish(msg);
        }

        // Use the position and orientation provided by the ros transform.
        centerLocation.x = odomPose.pose.position.x; //set centerLocation in odom frame
        centerLocation.y = odomPose.pose.position.y;


    }
}
```