Team 1:

Indoor Multi-camera Tracking System

Critical Design Review

Industry Advisor: B.Eng. Lui Tai
Supervisor: Prof. Costas Armenakis

December 04, 2017

Kevin Arindaeng (213094016)
Ariel Laboriante (212951984)
Zhuolin (Jack) Lu (212848834)
Varsha Ragavendran (213193065)

Link to Video:
https://youtu.be/u9b5l6OfDpA

TABLE OF CONTENTS

1. Scope

The goal of this project is to provide real-time navigation for autonomous mobile robots used in shipping warehouses. The U-Trackr system will fulfill the need for employee safety, increased productivity, and reduced company expenses allocated to damaged robots. Our key objective is to locate the position of the robot vehicle indoors, where GPS is unreliable, using only the image sequences of a limited camera system. As we build on our prototype, we aim to improve our system to also locate surrounding obstacles, predict the path of any collisions, notify the vehicle of any oncoming collisions (through vibrations or sound), and to predict the correct path to avoid collision. Our design includes the hardware, framing, and software of our camera-based system. However, the automation and design of the robot vehicle are outside of our system. We are assuming that our customers would be utilizing our system for their own robots. Therefore, the control of the robot's trajectory is beyond the scope of our project.

2. Testing Process and Requirements

Before prototyping, our team set out the vision of the prototype, outlining the functionalities of the system, the factors to consider, and a breakdown based on the design developed in the Preliminary Design Review document. Our team took an iterative approach to split our objectives into smaller states.

Thus, the prototype phase of this project was split into three solutions:

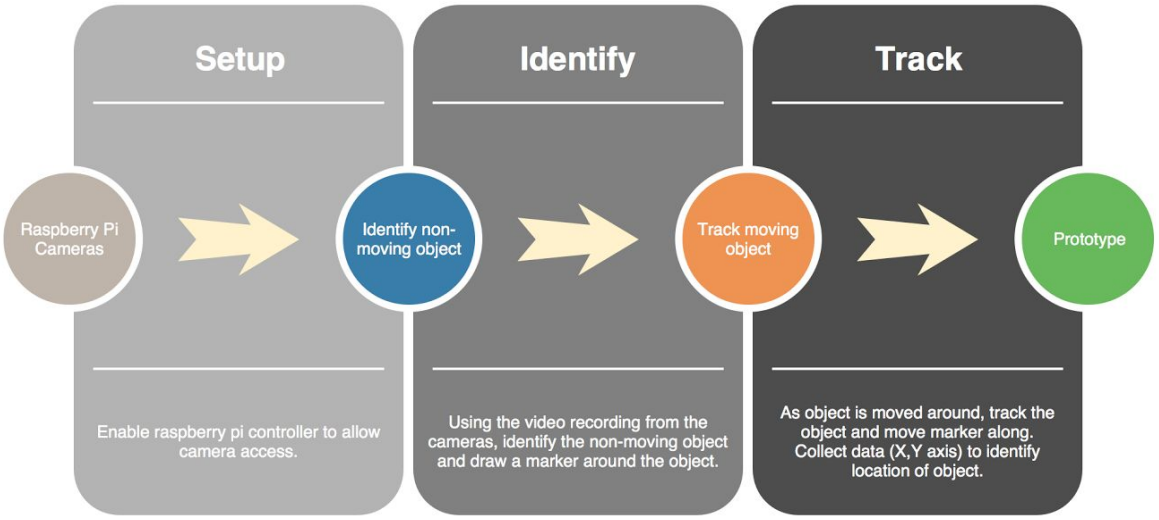

Figure 1. Solution scopes: 1. Setup, 2. Identify, 3. Track

2.1 Setup

In this phase of the prototype, our goal was to have the Raspberry Pi camera working with the microcontroller. We were unsure whether the video feed that would be received to our tracking program would have any significant impact to our system in terms of time delays.

In order to verify that such an issue is not present, we set up a server that could display the video recordings of the camera at real time. The cameras were set up, and programmed to record the video and feed the videos at real time to the server. By feeding to the server, we were able to verify that the feed of the camera to the server was almost close to real time.

2.2 Identifying an object of a particular color

The objective of this phase was to identify an object from each frame of the camera recording and add a marker on the target object. Since we were using a tennis ball to test this phase, we programmed a script to analyze each frame from the recording and locate an object based on color. The purpose of the marker was for us to verify that the program was identifying the correct object. In our case, the marker on the ball was a form of verification that the program was executing correctly.

2.3 Tracking the object and identifying the precise location

In this phase, our goal was to have our system track the object as it is moved around within the area of coverage of the cameras. To meet the requirements of this object, we programmed our scripts to collect data of the object's position with respect to the X and Y axis. As we changed the object's position, the marker on the target also moved along with it. The markers were then used to collect the data. The feed to the server allowed for us to verify that the marker moved as the ball moved. Therefore, the program was executing correctly and we were able to track an object in motion.

3. System Block Diagram

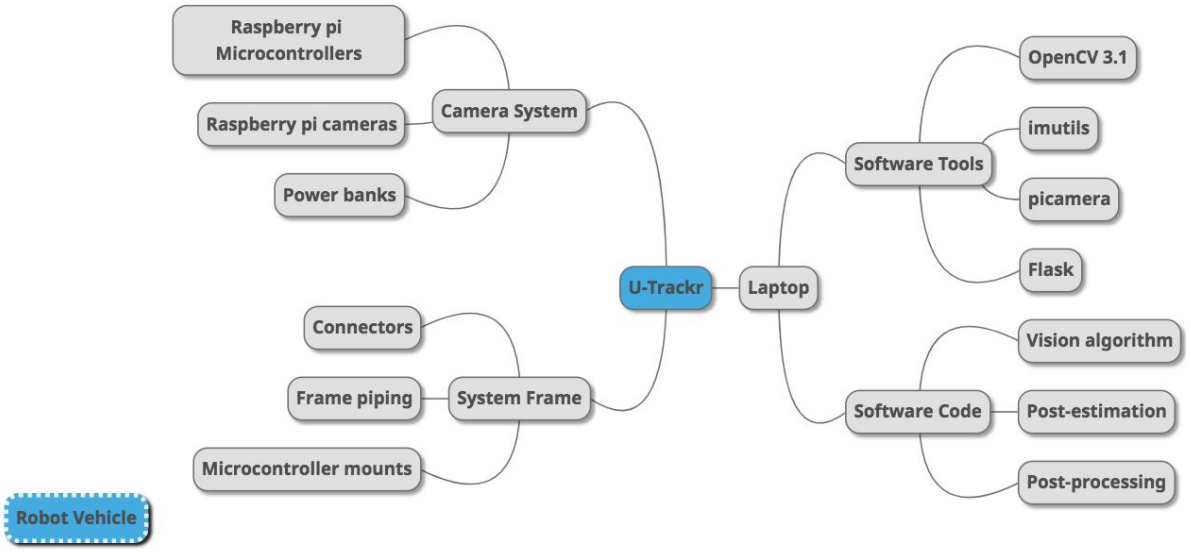

Figure 2. System diagram of the indoor multi-camera tracking system. *Robot vehicle is outside the system.

4. Critical Analysis

4.1 Addressing questions from PDR

How can we determine the variables such as the camera angle, height, the frame of axis and the position of the camera system?

While prototyping, we realized that the placement of all four cameras were not on the same height level. We did not account for this factor, and placed all four cameras on the frame, based on what we thought was the same level. To resolve this issue, we could attach a displacement sensor to each microcontroller in order to identify and verify that all four cameras are at the same height, and are consistent. The angles of the cameras were also not accounted for, therefore there exists a slight variation between the recordings from one camera to another. However, the Raspberry Pi Zero modules that were purchased came with a case that allows for the cameras to be placed within. This could help resolve this issue provided the cameras are all placed at the same height.

How can we track motion and how do we translate the movement into usable data?

Our prototype currently identifies an object based on the color of the object. As this object moves within the area the camera covers, a script has been programmed to track this object per frame. Therefore, as displayed in our prototype video, a marker on the object moves as this object moves. The X and Y axis of the object at starting position and object at ending position is recorded. The system will be further expanded to also record the Z axis position to support other uses. This data can be used when modelling the movement of the object and predicting collision.

Do we need a main microcontroller? And does it have to be in the center of the frame?

Our prototype currently does not have a main microcontroller in the center of the frame as was designed in the Preliminary Design Review document. We initially included the main microcontroller to manage all four cameras to retrieve and send the video recordings at a consistent pace. While prototyping with just four cameras, where each camera was sending its own recording to the server, we realized that the processors of the Raspberry Pis were a lot slower. The more tasks programmed within the controller, the slower the processing power of the Raspberry Pi. This also affects the rate at which the recordings are fed to the server. As displayed in the prototype video, there exists a delay due to this. Therefore, we might need a main microcontroller to act as the system's task manager and reduce the overhead.

How important is the camera frame rate important for image-processing?

This is the most critical part of our project. Since we are predicting if a collision will occur between two objects, it is important to have a fast camera frame rate, which will allow for images to be processed quickly, and model to predict collision. The camera frame rate of the Raspberry Pi is not great and is a limitation, but it is sufficient for the prototype phase.

How can we make our system portable?

Our prototype currently supports portability. Frames have been assembled using removable connectors. This ensures that the area for the coverage by cameras is consistent at all times.

4.2 Strengths, limitations, and recommendations for improvements

During the prototype phase, we realized that our design from the PDR is not entirely sufficient to have a well-rounded system, as expected. Although a couple of strengths were identified from the prototype, several limitations have arised. These limitations allow us to reconsider factors when implementing and developing the final system.

4.2.1 Strengths

Program has the ability to determine the location of the target object quickly based on color.

Provided there are no other objects with the same color, our system is able to quickly identify the target object. We considered this to be one of the strengths of the project as quick identification speeds up the initial process and collects the required data quickly to provide a real-time prediction of collision.

Portable frames which can be easily assembled to ensure area of coverage is same throughout the process of implementation and develop.

The process behind building the frame was to ensure that the system is portable and easily fixable at different locations. This is also considered to be one of the strengths of the project as it allows our system to be adaptable to various environments.

4.2.2 Limitations

Lag/Delay by approximately 2-5 seconds when feeding video to the server.

This factor is mainly due to the processor of the Raspberry Pi. Since several tasks were being accomplished by the controller, the processor speed decreases thereby causing a lag when the video recording was being fed to the server. Another possible source of delay could be from environmental factors such as the overload usage of WiFi.

The quality of the videos, in terms of resolution and frame rate, is low. The coverage range of this camera is also not sufficient for our project.

Depending on the processor, the video can have low pixel quality and resolution. This affects our system in terms of identifying the position of the target object(s) and directly affects lag and delay. The camera's limited field of view (horizontal and vertical) affects the reference space on which our system is defined.

The RGB Raspberry Pi cameras cannot detect objects under poor or low lighting.

Like any other cameras, the RGB Raspberry Pi camera module cannot detect objects in the dark. Therefore, the system is light dependent and restricted by low light setting.

As more tasks are programmed onto the microcontroller, the processing power decreases, thereby affecting the rate of data transmission, thus causing lag.
As mentioned earlier, the main microcontroller was not implemented, but it was initially intended to synchronize all four cameras and feed video recordings from each camera to the server. Each microcontroller currently tries to accomplish this task on its own, which affects the processor speed and causes delays.

Currently our prototype only supports detection of one object in motion. This is to be further improved down the road to support at least two if not more objects in motion, in order to predict collision.
The system only identifies one target object based on color. In order to predict collision between at least two or more objects, the system should be able to identify more than one object. This is part of the scope of the project which has not been implemented in the prototype.

Prototype does not differentiate between objects. Currently prototype identified objects based on color. This is to be further improved down the road, in order to be able to different between objects and/or people.
System has not been programmed to differentiate between objects. This feature would be useful if the project's scope was further expanded to communicate with the object in motion to correct its path, in order to prevent collision between this object and other environmental constraints, objects, or people.

The regulatory requirements of the system complying with the workplace safety and insurance board (WSIB) policies, and the Worker Health and Safety - Ontario Ministry of Labour policies, restricted our prototype to be performed with an inanimate object (tennis ball).
The purpose of our system is to track objects in motion and predict collision within a manufacturing plant. Due to the regulatory requirements identified from the Preliminary Document Review, our prototype was restricted to testing a tennis ball.

4.2.3 Recommendations for improvements

- We need to have a consistent field of view for the cameras, and to retrieve accurate coordinates, all cameras have to be placed at the same height level on the frame. Displacement sensors can be attached to each microcontroller to detect the height and distance between each microcontroller.
- The variation in delay with feeding the video to the server was mainly due to using various versions of the Raspberry Pi. We realized that the processor on the Raspberry Pi 2 Model B is a lot better and supports real time (~2second delay) with regards to tracking a moving object. Therefore, it might be necessary to be consistent across the board with either Raspberry Pi 2 *Model B* modules or Raspberry Pi Zero modules.

- Although the Raspberry Pi camera is sufficient for the prototyping phase, a camera with better resolution and frame rate will provide accuracy and less time delay with regards to covering more area, identifying precise location of objects, and feeding the videos to the server. Also, a camera that could operate and detect objects in the dark would be a favorable factor to consider.
- The Raspberry Pi currently detects objects based on color, and future improvements should include differentiating between objects and/or people. This would broaden the scope to support various objects and will not have a conflict when two objects of the same color are nearby.

4.3 Changes to the PDR plan as a result of prototyping activity

There have been no changes with regards to the scope of the system. However, to produce accurate results, our team will be executing most of the recommendations stated previously. This includes incorporating the use of displacement sensors to ensure that all the cameras are placed at the same height on the frame, staying consistent across the board by using the same model of the Raspberry Pi, and researching for better cameras that will provide a better resolution and frame rate.

4.4 Bottlenecks

There are currently four bottlenecks with the project:

- The microcontroller's processing power must ensure little to no delay when feeding the video recordings to the server and carrying out multiple tasks on the same controller.
- Synchronizing all four cameras to determine the accurate location of the objects
- Performing position calculation on images to determine precise location of the objects
- Modelling these objects to determine and prevent collision, if the objects are very close

The key question we need to resolve is how to program with OpenCV to analyze all recordings from all four cameras to pinpoint the exact location of the objects. Each camera would have to feed the video recording to the server with a timestamp, which allows for OpenCV to compare this video recording with recordings from the other cameras. We can then performing position calculation on the images from various angles to pinpoint the exact location of the objects. Since the team has not worked with OpenCV before, applying position calculations on images being processed using OpenCV is a hurdle to overcome.

4.4 Questions generated during prototyping

- How will the program react if more than one object is present in the area covered by the camera, i.e. if more than one object is in motion, how will the system know which object to track?
- If the camera is able to operate and view objects in motions in the dark, how will it detect an object of a particular color?

- What is the approximate time estimate between the start time of the video recording from the camera to when the modelling in real-time predicts collision?

5. Detailed Theory

Tracking systems are crucial because they provide a way of finding the position of an object and deriving the motion and direction. Our system is needed when it comes to obtaining the real-time location of a 3-dimensional object in a 2-dimensional image plane. The image-based tracking system developed in this project is comprised of the multi-camera system, computer, microcontrollers, and processing software. In this section, the theory, the methodology behind the vision algorithm, the processing software, and the indoor multi-camera tracking system will be explained in detail.

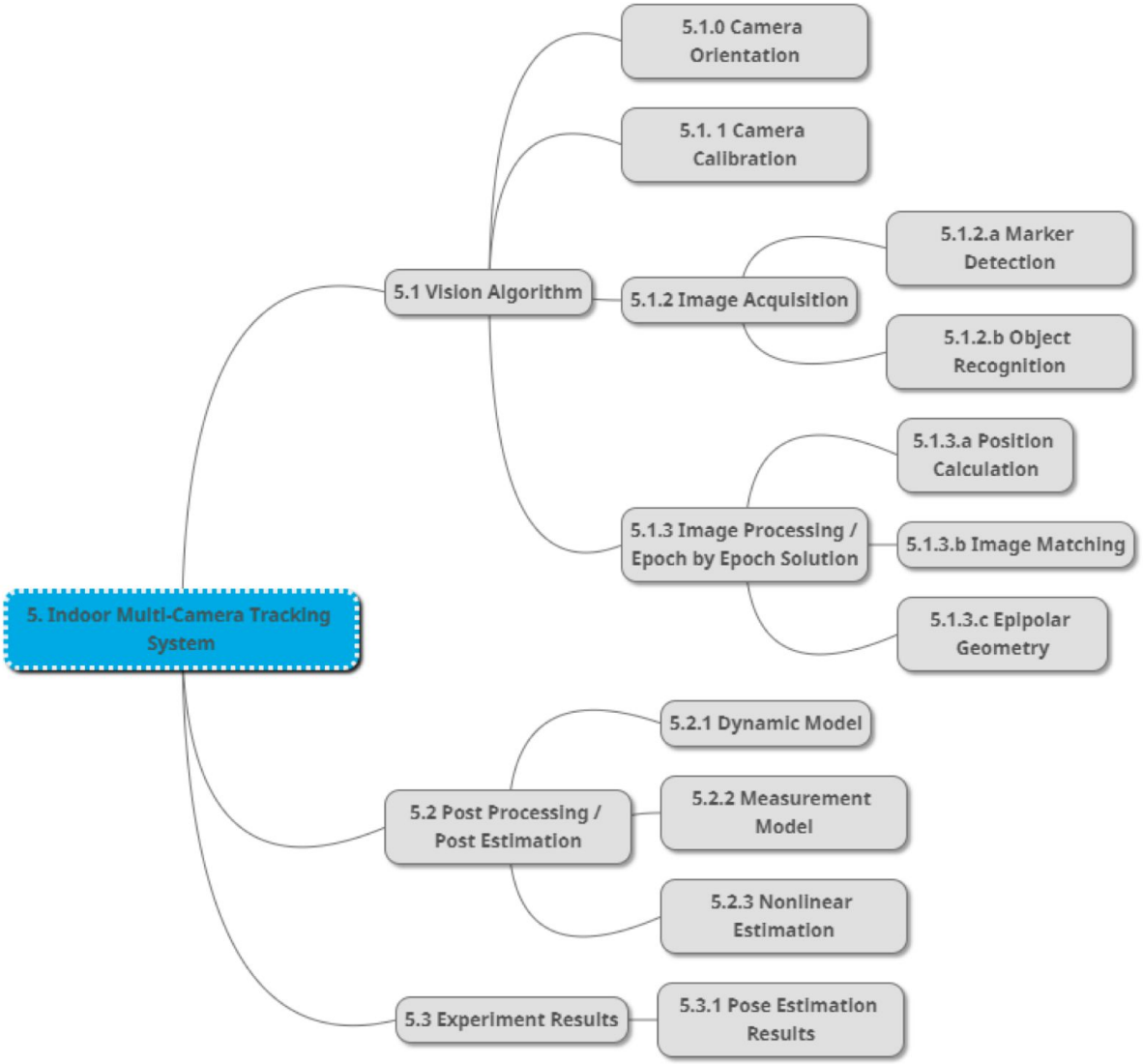

Figure 3. Multi-camera Tracking System: The theories related to each aspect of the definition space.

5.1 Vision algorithm

This section presents the vision algorithm for the pre- and post-image processing of the camera system as well as the implementation of the cameras network and the bottleneck technology restricting the indoor multi-camera tracking system.

5.1.1 Camera orientation

The multi-camera tracking system consists of four Raspberry Pi camera v2 modules installed at each corner of the frame and attached with the Raspberry Pi microcontrollers. The data acquisition and transmission of data is done through the microcontrollers to reach the main processing board of the computer. The frame of the indoor system has the dimensions 0.762 m X 0.762 m X 0.762 m (or 2.5 ft X 2.5 ft X 2.5 ft). The tilt angles of the cameras have been adjusted to 45 degrees, and the camera itself has a 62.2 degrees horizontal and 48.8 degrees vertical field of view (as per the specification of the camera attached in the Appendix). This way, each of the individual cameras can track all the motion within the defined space. The following diagram is a model of our camera system.

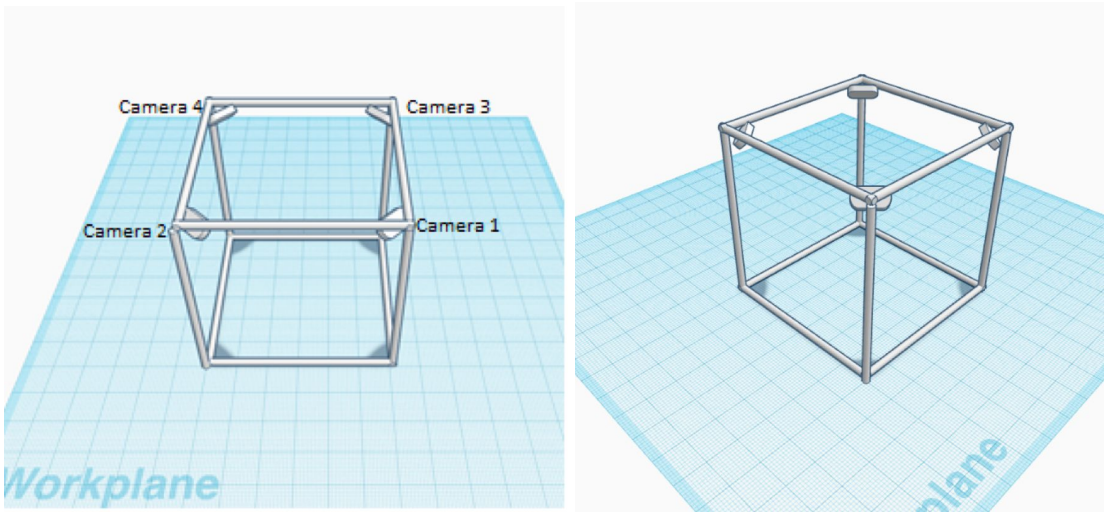

Figure 4. Designed Camera Network: The cameras placed on the frame provide an overdetermined system where the positioning of two cameras can be checked by other cameras through statistical analysis.

5.1.2 Camera model and camera calibration

The basic model of a pinhole camera describes the relationship between mapping in 3D world to 2D image. The 2D image is represented by a 3 by 4 homogeneous camera projection matrix with 11 degrees of freedom to correlate the 2D image to a 3D object. These parameters include three translations, three rotations, one principle point, two pixel dimensions and two skew parameters [3]. The following equation represents this relationship.

Mapping 3D coordinate to 2D	
$\bar{x}_{image} = P\bar{X}_{world}$	Where, $\bar{X}_{world}$ is the 3D world positioning element represented by coordinates vectors and scale factor depth W_s to form $(X, Y, Z, W_s)^T$ $\bar{x}_{image}$ is the 2D image point represented by a $(x, y, w_s)^T$ and w_s as the scale factor depth
$P = K[R_1^{cam}\|t_1^{cam}]$	P is the 3 by 4 homogeneous camera projection matrix R_1^{cam} is the rotation matrix from inertial frame center of the camera t_1^{cam} is the translation matrix from inertial frame center of the camera
And, $K = \begin{bmatrix} a_x & s & x_0 \\ 0 & a_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$	(a_x, a_y) is the pixel dimension (x_0, y_0) is the skew parameter s is the principle point
- Basic principle of pinhole camera equation	

The camera calibration procedure estimates the camera projection matrix. The calibration is assumed to be negligible and the cameras are assumed to be in factory quality (the specifications of the Raspberry Pi and Raspberry Pi camera module v2 are provided in the Appendix). In other words, there is no projection matrix caused by the camera system since in prototyping, the accuracy of the system does not matter as much. Alternative ways for camera calibration can be done using the calibration toolbox provided by the *MATLAB* database.

5.1.3 Image acquisition

The image acquisition of the project is considered to be the **bottleneck technology** since the input rate of image data has a direct correlation on the output position of the object being track. This part of the project is accomplished by first detecting the object colour marker to provide the 2D-image position, and then recognizing the object in the frame of reference.

5.1.3.a Marker detection

Marker detection is the extraction of distinct colours in the image given by the camera. The marker must be distinguishable and have different features than its surroundings. For this project, the software used to detect RGB colour space have a range of (256,256,256) for each colour. The colour selected for detection is in the range of green. Any other sources of green colour produced by the surrounding can affect the detection of the marker [4]. The use of yellow light is also not recommended as yellow light interferes with the selected marker colour and thus does not allow the OpenCV program to operate. Another issue due to lighting is when the room is too dark, the threshold DN value of green cannot be detect.

The DN value threshold refers to the maximum and minimum values that the program needs to detect the colour “green”.

This value is defined in the code by *greenLower = (29,86,6)* and *greenUpper = (64,255,255)*. Once the marker is detected, the object can be recognized, and the position of the object is defined in the 2D image space [4].

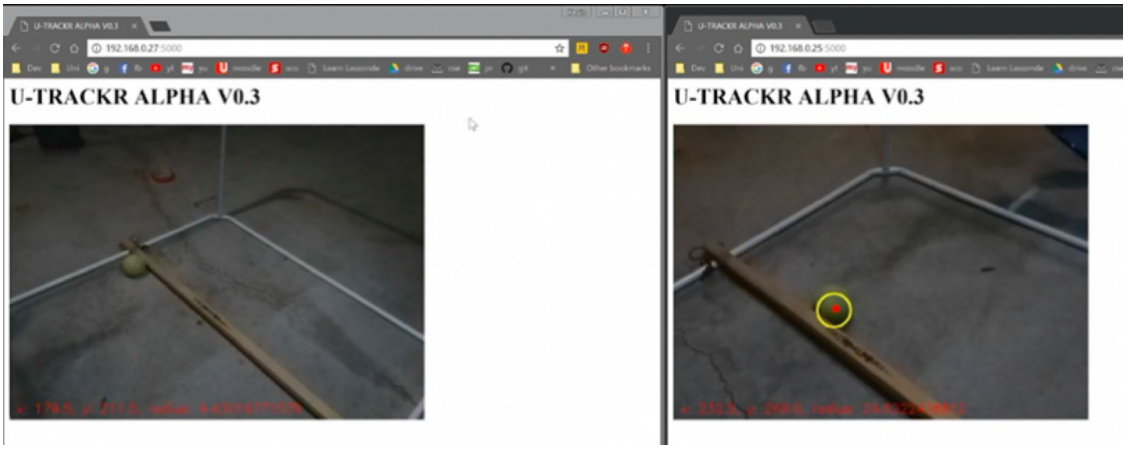

Figure 5. Marker Detection: The image on the left is not detected, and this can be interpreted two ways. First, the ball has a green DN value threshold outside the one previously stated. Second, the lighting caused the image on the right to appear in view of the camera and the ball was detected.

5.1.3.b Object recognition

The object recognition is the detection of the object’s colour marker and the provision of the positional coordinate of the object. The OpenCV program recognizes objects with the green colour marker. It creates a circle with a set radius around the object and a red mark at the centre of its position. The object being recognized does not have to be a circle, but in the case of this project, the centroid position of a circle is the easiest to program. The position of the object is defined with respect to the top left of the reference image, where it is set as coordinate (0,0). From there the center of the ball is defined and the output is created in the form of a .txt file. The .txt file allows for post-processing of the data to convert it to 3D real-world position [2].

Figure 6. Object Recognition: The tested tennis ball has a defined XY coordinate system with respect to the image frame. The set radius of the ball is defined.

SampleOutput - Notepad

File Edit Format View Help

```
time:2017-12-03 22:24:03.402, x: 523.636474609, y: 164.570999146, radius: 72.9933853149
time:2017-12-03 22:24:03.836, x: 63.0, y: 42.5, radius: 74.6074066162
time:2017-12-03 22:24:04.024, x: 524.286682129, y: 164.304397583, radius: 72.9604110718
time:2017-12-03 22:24:04.201, x: 522.925537109, y: 164.138839722, radius: 72.0964584351
time:2017-12-03 22:24:04.371, x: 520.373718262, y: 164.15788269, radius: 69.0024032593
time:2017-12-03 22:24:04.553, x: 524.269470215, y: 164.511260986, radius: 73.0487442017
time:2017-12-03 22:24:04.727, x: 521.522216797, y: 165.975265503, radius: 71.8767166138
time:2017-12-03 22:24:04.903, x: 524.195861816, y: 164.059738159, radius: 72.2697982788
```

Figure 7. Sample output of the system: The file is formatted with the time, position, and radius of the circle from the centroid.

5.1.4. Image processing of epoch by epoch solution

The image processing of the project defines the calculation of the image matching, position calculation and epipolar geometry. This allows for the transformation of 2D images into 3D real-world positions.

5.1.4.a Position calculation

The position calculation must be completed to transform 2D image space into 3D cartesian coordinates. This is done using several predefined parameters such as the position of the camera (h,d), the tilt angle of the webcam towards the horizontal α and the perpendicular distance to the webcam lens of the objects when it is captured dZ_0 [1].

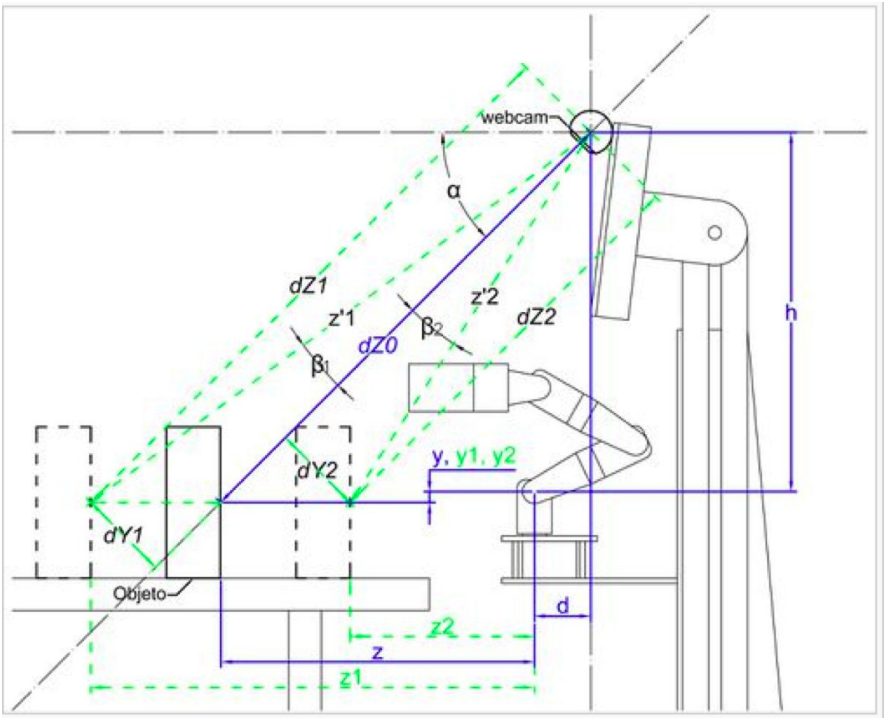

Figure 8. Sketched Position Calculation: Here the object is defined as a rectangular figure. In the project, the object is a circle.

Position calculation of a 2D image (1/2)	
$L_s = \sqrt{\left(E_{SD(x)} - E_{SI(x)}\right)^2 + \left(E_{SD(y)} - E_{SI(y)}\right)^2}$	Where, L_s is the object's top side width in the scene (in pixels) $E_{SD(x,y)}$ is the object's top right corner in the scene (in pixels) $E_{SI(x,y)}$ is the object's top left corner in the scene (in pixels) $E_{II(x,y)}$ is the object's bottom left corner in the scene (in pixels) $E_{ID(x,y)}$ is the object's bottom left corner in the scene (in pixels)
$L_I = \sqrt{\left(E_{ID(x)} - E_{SI(x)}\right)^2 + \left(E_{ID(y)} - E_{SI(y)}\right)^2}$	L_s is the object's bottom side width in the scene (in pixels)
$H_I = \sqrt{\left(E_{II(x)} - E_{SI(x)}\right)^2 + \left(E_{II(y)} - E_{SI(y)}\right)^2}$	H_I is the object's left side height in the scene (in pixels)
$H_D = \sqrt{\left(E_{ID(x)} - E_{SD(x)}\right)^2 + \left(E_{ID(y)} - E_{SD(y)}\right)^2}$	H_D is the object's right-side height in the scene (in pixels)
$dArea = \left(H_{image} * L_{image}\right) - \left(\frac{H_I + H_D}{2} * \frac{L_S + L_I}{2}\right)$	$dArea$ is the area of the object surface H_{image} is the image height in pixels L_{image} is the image width in pixels
$dZ = dZ_0 + \frac{dArea}{38}$	dZ is the angle of field of view
$dX = \frac{\left(\frac{640}{2} - \left(\frac{E_{SI(x)} + E_{II(x)}}{2} + \frac{L_S + L_I}{4}\right)\right) * dZ}{585}$	dX is the X coordinate of the object in pixel space
$dY = \frac{\left(\frac{640}{2} - \left(\frac{E_{SI(y)} + E_{II(y)}}{2} + \frac{H_S + H_I}{4}\right)\right) * dZ}{585}$	dY is the Y coordinate of the object in pixel space
$\beta = \tan^{-1}\left(\frac{Y}{dZ}\right)$	β is the angle between bottom of object top of object in pixel space
$x = DX$	x is the X coordinate in 2D image space

- Position calculations and parameters. Note dX, dY, dZ are the pixel space coordinate and x, y are the 2D real-world cartesian coordinates, with z being the distance between frame and object.

Position calculation of a 2D image (2/2)	
$y = h - \frac{dZ}{cos\beta} * \sin(\alpha - \beta)$ $z = h - \frac{dZ}{cos\beta} * \sin(\alpha - \beta) - d$	Where y is the X coordinate in 2D image space h is the height of the camera α is the angle between the camera and the horizontal plane z is the distance between the distance of the frame to object d is the distance between the frame and the camera
- Note dX, dY, dZ are the pixel space coordinate and x, y are the 2D real-world cartesian coordinates, with z being the distance between frame and object.	

5.1.4.b Image matching

Image matching is the process of selecting a matching entity from one image to find the conjugate entity in another overlapping image. There are two common methods of image matching: area-based matching (ABM) and feature-based matching (FBM). The area-based matching is the process of matching gray level distributions in a small area of two stereo-paired images. The similarity measure between image patches can be computed using the cross-correlation coefficient and least square matching of 1D and 2D. From there, the template window created from the first image (which has the similarity measures) is matched with the search window of the second image to find that area on the second image[2].

Figure 9. Area-Based Image Matching: The matching of template from one image with similar measure to the second image within the search window.

Feature-based matching is the extraction of features such as points or line from the images being compared. The feature similarities can be defined by the shape, size, length, curvature and the gradient across the edge of an image [2]. Feature-based matching is usually used prior to the earlier stage of image processing or before the detection of the markers whereas area-based matching is used after object recognition and marker detection where a template search window can be created from the recognized object. Since image matching is not part of the bottleneck technology as it requires images first, this part was not implemented in the *Critical Design Review*.

5.1.4.c Epipolar geometry

The epipolar geometry is the geometry between two cameras which consists of an epipole e' (the point of intersection of the line joining the camera center), an epipolar plane H_{π} (a plane containing the baseline), an epipolar line l' (the intersection of an epipolar plane with the image plane). This relationship is demonstrated by the figure below.

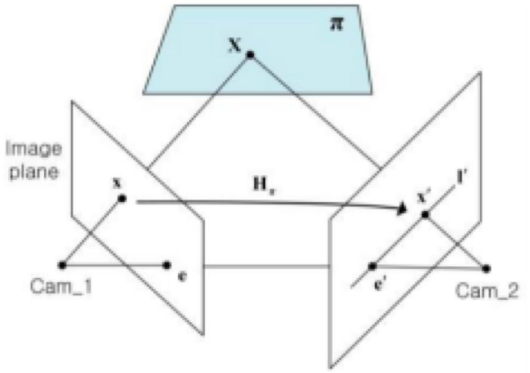

Figure 10. Epipolar Geometry Relationships: Note that the two images must have matching features or area to perform epipolar geometry.

The red line formed with the wooden stick both image shows the relationship of similar feature and geometry needed to epipolar calculation.

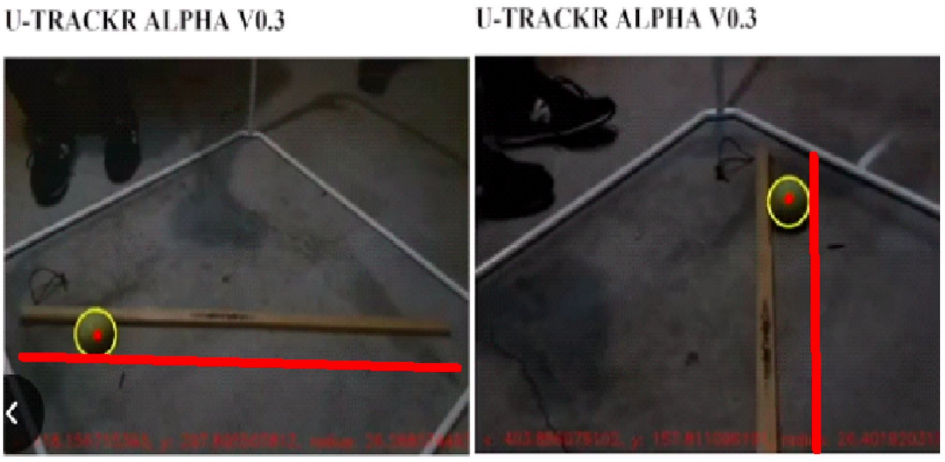

Figure 11. Epipolar Geometry of Tennis Ball Feature: Red line indicating the epipolar line.

The epipolar geometry used for tracking corresponds to the least square adjustment condition. The condition states that the blenders are minimized to calculate 3D position [3].

Epipolar Geometry for Tracking	
$x'^T F x = 0$ $F = \begin{bmatrix} x_{image} \\ y_{image} \\ 1 \end{bmatrix}$	Where, F is the fundamental model for matrix representing 2D mapping x' *and* x are the epipolar points
- Note this is the linear mapping model of 2D image	

5.2 Post-processing and post-estimation

The post-processing and post-estimation of the project analysis is done on the final output results of each camera and through series of statistical test, the correlation and accuracy of the network can be determined. This part of the project is not a bottleneck technology, so it was not included in our prototype. The indoor tracking system only works with two Raspberry Pi camera modules so an overdetermined network cannot be accomplished. Thus, the post-processing and post-estimation cannot be done due to too many unknown variables.

5.3 Experimental results

The following is a sample demonstration of the tracking motion and position calculation over time. The raw data can be found in the Appendix.

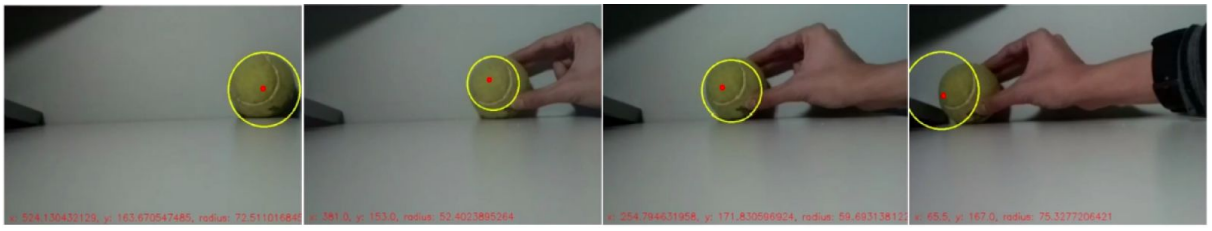

Figure 12. Motion of A Ball rolling: Note the output (x,y) coordinates are given in .txt file.

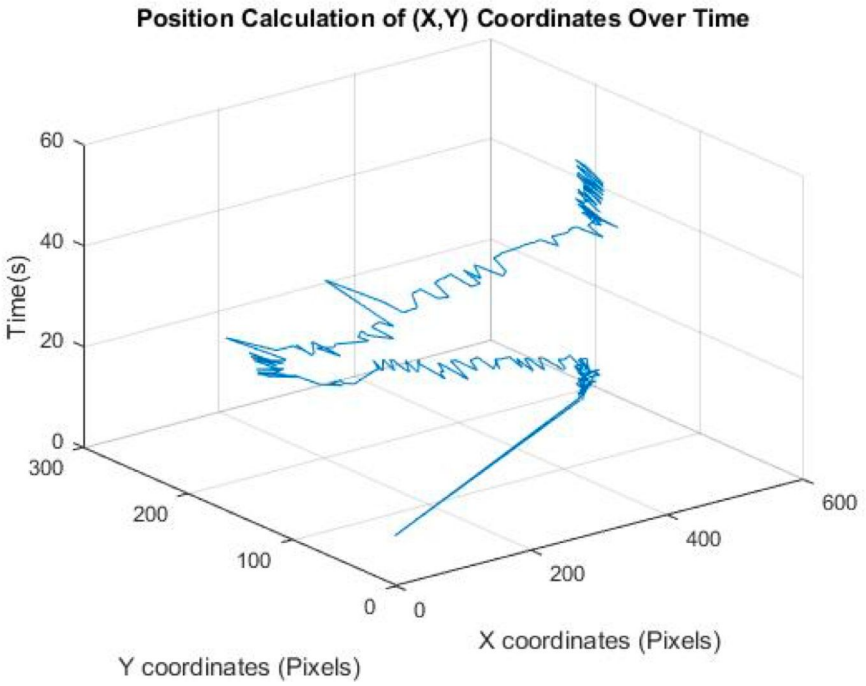

Figure 13. Position Calculation of (X, Y) Coordinates Over Time: The graph shows the motion over time of the tennis ball in one and a half cycles or from one end to the other and back.

6. Refined Project Schedule

A breakdown of scheduled tasks and resources can be found below. This can also be seen using Microsoft Project with various views by following the link.

	Resource Name	Type	Material	Initials
1	**Kevin**	**Work**		**K**
2	**Ariel**	**Work**		**A**
3	**Varsha**	**Work**		**V**
4	**Jack**	**Work**		**J**
5	Microcontrollers	Material		M
6	Cameras	Material		C
7	Autonomous Rovers	Material		A
8	CDR Prototype	Material		C

Figure 14. Resource Allocation

	Task Name	Grade Percentage Remaining	Duration	Start	Finish	Predec	Resource Names
1	**U-TRACKR**	**65%**	**119 days**	**Tue 12/5/17**	**Mon 4/2/18**		
2	**Test Readiness Review (TRR)**	**10%**	**63 days**	**Tue 12/5/17**	**Mon 2/5/18**		
3	Obtain new microcontrollers, cameras, and rover parts		15 days	Tue 12/5/17	Tue 12/19/17		Kevin
4	Meeting with industry advisor to discuss goals and feedback from the CDR		1 day	Mon 12/11/17	Mon 12/11/17		Ariel,Jack,Kevin,Varsha
5	Ensure the new microcontrollers and cameras can interface correctly		4 days	Tue 12/19/17	Sat 12/23/17	3	Kevin,Varsha,Cameras[1],Microcontrollers[1]
6	Convert prototype functionality into the new microcontrollers and cameras		10 days	Sat 12/23/17	Tue 1/2/18	5	Kevin,Varsha,Cameras[1],CDR Prototype[1],Microcontrollers[1]
7	Get position calculation working on the microcontroller		14 days	Wed 1/3/18	Tue 1/16/18	6	Ariel,Varsha,Jack,Kevin,Cameras[1],Microcontrollers[1]
8	Build both autonomous vehicles		14 days	Tue 12/19/17	Tue 1/2/18	3	Ariel,Jack,Autonomous Rovers[1]
9	Have rovers move in a straight line autonomously		2 days	Wed 1/3/18	Thu 1/4/18	8	Ariel,Jack,Autonomous Rovers[1]
10	Have rovers be able to turn autonomously		2 days	Fri 1/5/18	Sat 1/6/18	9	Ariel,Jack,Autonomous Rovers[1]
11	Interface the rovers with the microcontrollers		8 days	Sun 1/7/18	Sun 1/14/18	6,10	Jack,Varsha,Autonomous Rovers[1],Cameras[1],Microcontrollers[1]
12	Create a UI application to interface with the microcontroller settings and data		8 days	Wed 1/3/18	Wed 1/10/18	6	Kevin,Microcontrollers[1],Cameras[1]
13	Set up an alert mechanism on the UI application in a collision event		8 days	Thu 1/11/18	Thu 1/18/18	12	Kevin,Autonomous Rovers[1],Cameras[1],Microcontrollers[1]
14	Prevent the rovers from colliding in a collision event		12 days	Fri 1/19/18	Tue 1/30/18	11,13	Ariel,Jack,Kevin,Varsha,Autonomous Rovers[1],Cameras[1],Microcontrollers[1]
15	Meeting with the technical supervisor to discuss design goals and improvements		2 days	Wed 1/31/18	Thu 2/1/18	14	Ariel,Jack,Kevin,Varsha
16	TRR Final Report		6 days	Wed 1/31/18	Mon 2/5/18	14	Ariel,Jack,Kevin,Varsha
17	**Test Review (TR) (Rough Estimate)**	**10%**	**21 days**	**Tue 2/6/18**	**Mon 2/26/18**	**2**	
18	Unit Testing		6 days	Tue 2/6/18	Sun 2/11/18		
19	Integration Testing & Continuous Refinement		5 days	Mon 2/12/18	Fri 2/16/18	18	
20	Acceptance Testing		5 days	Sun 2/18/18	Thu 2/22/18	19	
21	TR Draft		2 days	Fri 2/23/18	Sat 2/24/18	20	
22	TR Final Report		2 days	Sun 2/25/18	Mon 2/26/18	21	
23	**Final Formal Documentation (FFD) (Rough Estimate)**	**25%**	**14 days**	**Tue 2/27/18**	**Mon 3/12/18**	**17**	
24	FPD Draft		7 days	Tue 2/27/18	Mon 3/5/18		
25	FPD Final Report		7 days	Tue 3/6/18	Mon 3/12/18	24	
26	**Product Release Presentation/Exhibits (Rough Estimate)**	**10%**	**20 days**	**Wed 3/14/18**	**Mon 4/2/18**	**23**	
27	Demo Materials Preparation (Bristol Boards, Videos, etc)		10 days	Wed 3/14/18	Fri 3/23/18		
28	Demo Practice Preparation		9 days	Fri 3/23/18	Sun 4/1/18	27	
29	Demonstration		1 day	Mon 4/2/18	Mon 4/2/18	28	
30	**In-class Feedback**	**5%**					
31	**Weekly Reports (On-going)**	**5%**					

Figure 15. Gantt Chart View of the Project Schedule

6.1 Risks

Some of the associated risks that may happen throughout the life-cycle of the project within the Test Readiness Review (TRR) are as follows:

Delay in the shipment of the new expenses: Since this is the major predecessor of all of the major tasks, this has the risk to delay most of the project.

Faulty equipment ordered: If the microcontrollers are not able to interface as they are expected to, it would delay the development of major features such as the position calculation and the autonomous rover.

Possible damage due to collision testing: While developing the feature to prevent collision events, collisions are expected to occur on the autonomous rovers, causing possible equipment damage.

Possible loss of functionality when converting the original prototype to the new microcontrollers: The new microcontrollers may interface differently with the camera modules, causing some features from the prototype developed on the CDR to not work. This may require reimplementation which has the risk of delaying the project.

7. Adjusted Expenses Tables

Table 1: TRR Expenses Table for U-TRACKR

Deliverable Area	Item (Problem Space)	Item (Solution Space)	Estimated Cost	Actual Cost	Description
TRR	Microcontroller (x4)	BeagleBone Black (x4)	($87.00 * 4) = $348	TBD	Controls and processes information from the camera modules. Needs to have strong processing power to read and write data quickly to as server.
	Cameras (x4)	Pixy CMUcam5 (x4)	($94.48 * 4) = $377.96	TBD	Takes high-definition videos in real time for vision based tracking. Needs to have high frame rate to keep up with photogrammetry calculations.
	Programmable Land Rover (x2)	Adafruit DC & Stepper Motor HAT for Raspberry Pi	($29.99 * 2) = $59.98	TBD	Used as a payload to track, as well as prevent collisions towards each other. The previous Raspberry Pi microcontrollers will be used to create this.
		Brass M2.5 Standoffs for Pi HATs - Black Plated - Pack of 2 (x2)	($6.49 * 2) = $12.98	TBD	
		Mini Robot Rover Chassis Kit - 2WD with DC Motors	($42.99 * 2) = $85.98	TBD	
TRR BUDGET = $750 TRR TOTAL ESTIMATED EXPENSES = $884.90 **TRR ACTUAL EXPENSES: TBD**					

Table 2: Updated CDR Expenses Table for U-TRACKR

Deliverable Area	Item (Problem Space)	Item (Solution Space)	Estimated Cost	Actual Cost	Description
CDR (Prototype)	Microcontroller (x4)	~~Raspberry Pi Zero (x2)~~ Raspberry Pi Zero W Starter Kit with Case (x2)	($34.99 * 2) = $69.98	($49.99 * 2) = **$99.98**	Controls and processes information from the camera modules. (Two Raspberry Pi modules are provided by the team for prototyping)
	Cameras (x4)	Raspberry Pi Camera Module V2 (x4)	($30.99 * 4) = $123.96	($30.99 * 4) = **$123.96**	Takes high-definition videos in real time for vision based tracking.
	Frame	~~PVC Pipes and Fittings~~ *PVC Pipes*	($8.73 * 2) = $17.46	**$26.40**	Holds the camera modules in opposite and equidistant positions.
		T-Connectors (x8)	-	**$15.04**	
		45° Connectors (x16)	-	**$25.28**	
		Dust Masks	-	**$2.50**	
		Spray Paint	-	**$11.99**	
CDR BUDGET = $250 CDR TOTAL ESTIMATED EXPENSES = $211.40 **CDR ACTUAL EXPENSES: $305.15**					

Table 3: Updated Common Expenses Table for U-TRACKR

Deliverable Area	Item (Problem Space)	Item (Solution Space)	Estimated Cost	Actual Cost	Description
All Deliverable Areas	Primary Processing System	Personal Laptop	-	-	Tracks, locates, and models the moving object(s) programmatically using image sequences. (Provided by the team for prototyping)
	Portable Power Supply for Microcontrollers (x4)	USB Power Banks	-	-	Provides an external power source for the microcontroller. (Provided by the team for prototyping)
COMMON TOTAL ESTIMATED EXPENSES = $0 **COMMON ACTUAL EXPENSES: $0**					

8. Self-Evaluation of Critical Design Review (Rubric)

RUBRIC CRITERION FOR CDR	SELF-EVALUATION RANKING	OUR JUSTIFICATIONS
Apply an iterative process to refine or assign solutions for a given engineering design problem.	Meeting criterion (3/4): Applies an appropriate number of iterations to refine or assign solutions for a given engineering design problem.	The objective of the prototype was broken down into smaller portions to accomplish the chosen engineering design problem.
Integrate design subsystems into a complete system.	Meeting criterion (3/4): Elegant integration of design subsystems into a complete system.	All subsystems have been identified in the block diagram and the video describes the building process in great detail.
Justify the strength and limitations of the solution and make recommendation for possible improvements.	Exceeding criterion (4/4): Critical evaluation of the strengths and limitations of the solution; makes effective recommendations for possible improvements.	Every strength and limitation of the prototype identified has a clear rationale behind the category. Recommendations have been provided to improve the project based on factors such as time, efficiency, etc.

Incorporate appropriate considerations of ethical, social, environmental, legal, and regulatory factors Into an engineering design.	Meeting criterion (3/4): Incorporates key appropriate and adequate consideration of ethical, social, environmental, legal, and regulatory factors into an engineering design.	Regulatory requirements identified in the PDR system requirements section has been thought of while developing the prototype. Furthermore, new environmental constraints have been identified through the process of developing the prototype.
Develop concise and coherent reports and design documents that reflect critical analysis and synthesis.	Exceeding criterion (4/4): Writes concise and coherent reports and design documents; presents critical analysis and synthesis of information in a complete and compelling manner; reports and documents have a polished professional appearance	The report is written for the purpose of documenting the prototype and the associated bottlenecks. The presentation of the critical analysis and synthesis is professional and well-polished.
Incorporate principles of sustainable design/development in an engineering task.	Exceeding criterion (4/4): Fully incorporates principles of sustainable design/development in an engineering task	The project demonstrates sustainable design throughout the report. We have consulted with numerous reliable sources such as the industry supervisor, the project supervisor, and Geomatics Master students.
Adjust project schedule based on project status.	Meeting criterion (3/4): Makes appropriate adjustments to a project schedule based on project status	The project schedule takes into account the feedback from the PDR, and also utilizes the resources used within the project.
Monitor risks during the life-cycle of the Project.	Meeting criterion (3/4): Monitors key risks during the life-cycle of a project	The main key risks within the Test Readiness Review phase are identified, with reasonings behind each risk.
Apply engineering concepts and fundamentals, theories and practices to solve real-world open-ended engineering problems.	Exceeding criterion (4/4): Applies a wide range of engineering concepts and fundamentals, theories and practices to solve real- world open-ended engineering problems	The project is supported with the theory and knowledge behind vision algorithm, image-processing. Our supervisor provided us with papers on motion tracking and photogrammetric processing.
Use specialized engineering knowledge of design specific components, systems or processes to solve engineering problems.	Exceeding criterion (4/4): Sophisticated use of specialized engineering knowledge of design specific components, systems or processes to solve engineering problems	The knowledge and design of the specific component have been researched thoroughly through iterations of consultation with industry adviser, project supervisor, and research studies done by numerous universities around the world.

9. References

1. *Objects recognition and position calculation (Webcam)*, robotica.unileon.es/index.php/Objects_recognition_and_position_calculation_(webcam).

2. Costas , Armenakis. “ESSE 4640 Digital Terrain Modelling Lecture 11.” 1 Dec. 2017, Toronto. Slides 34-60

3. Oh, Hyondong, et al. “Indoor UAV Control Using Multi-Camera Visual Feedback.” *Unmanned Aerial Vehicles*, 2010, pp. 57–84., doi:10.1007/978-94-007-1110-5_6.

4. “OpenCV Track Object Movement.” *PyImageSearch*, 21 Sept. 2015, www.pyimagesearch.com/2015/09/21/opencv-track-object-movement/.

10. Appendix

Raspberry Pi *Camera Module Specifications*

	Camera Module v1	**Camera Module v2**
Net price	$25	$25
Size	Around 25 × 24 × 9 mm	
Weight	3g	3g
Still resolution	5 Megapixels	8 Megapixels
Video modes	1080p30, 720p60 and 640 × 480p60/90	1080p30, 720p60 and 640 × 480p60/90
Linux integration	V4L2 driver available	V4L2 driver available
C programming API	OpenMAX IL and others available	OpenMAX IL and others available
Sensor	OmniVision OV5647	Sony IMX219
Sensor resolution	2592 × 1944 pixels	3280 × 2464 pixels
Sensor image area	3.76 × 2.74 mm	3.68 x 2.76 mm (4.6 mm diagonal)
Pixel size	1.4 µm × 1.4 µm	1.12 µm x 1.12 µm
Optical size	1/4"	1/4"
Full-frame SLR lens equivalent	35 mm	
S/N ratio	36 dB	
Dynamic range	67 dB @ 8x gain	
Sensitivity	680 mV/lux-sec	
Dark current	16 mV/sec @ 60 C	
Well capacity	4.3 Ke-	
Fixed focus	1 m to infinity	
Focal length	3.60 mm +/- 0.01	3.04 mm
Horizontal field of view	53.50 +/- 0.13 degrees	62.2 degrees
Vertical field of view	41.41 +/- 0.11 degrees	48.8 degrees
Focal ratio (F-Stop)	2.9	2.0

Figure 16. Raspberry Pi Camera Module: Implemented Module v2.

Technical Specifications

The Raspberry Pi Zero W extends the Pi Zero family. Launched at the end of February 2017, the Pi Zero W has all the functionality of the original Pi Zero, but comes with with added connectivity, consisting of:

- 802.11 b/g/n wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)

Like the Pi Zero, it also has:

- 1GHz, single-core CPU
- 512MB RAM
- Mini HDMI and USB On-The-Go ports
- Micro USB power
- HAT-compatible 40-pin header
- Composite video and reset headers
- CSI camera connector

Figure 17. Raspberry Pi Camera Module: Zero W

Sample .txt file output:

```
time:2017-12-03 22:24:03.402, x: 523.636474609, y: 164.570999146, radius: 72.9933853149
time:2017-12-03 22:24:03.836, x: 63.0, y: 42.5, radius: 74.6074066162
time:2017-12-03 22:24:04.024, x: 524.286682129, y: 164.304397583, radius: 72.9604110718
time:2017-12-03 22:24:04.201, x: 522.925537109, y: 164.138839722, radius: 72.0964584351
time:2017-12-03 22:24:04.371, x: 520.373718262, y: 164.15788269, radius: 69.0024032593
time:2017-12-03 22:24:04.553, x: 524.269470215, y: 164.511260986, radius: 73.0487442017
time:2017-12-03 22:24:04.727, x: 521.522216797, y: 165.975265503, radius: 71.8767166138
time:2017-12-03 22:24:04.903, x: 524.195861816, y: 164.059738159, radius: 72.2697982788
time:2017-12-03 22:24:05.075, x: 550.5, y: 171.0, radius: 47.9297332764
time:2017-12-03 22:24:05.243, x: 523.492431641, y: 164.884780884, radius: 72.5658950806
time:2017-12-03 22:24:05.423, x: 523.006225586, y: 164.363830566, radius: 71.8826522827
time:2017-12-03 22:24:05.602, x: 551.0, y: 171.0, radius: 47.7599143982
time:2017-12-03 22:24:05.784, x: 551.5, y: 168.5, radius: 49.9650878906
time:2017-12-03 22:24:05.954, x: 524.130432129, y: 163.670547485, radius: 72.5110168457
time:2017-12-03 22:24:06.131, x: 525.0, y: 169.5, radius: 66.9347686768
time:2017-12-03 22:24:06.310, x: 541.5, y: 172.0, radius: 56.9057235718
time:2017-12-03 22:24:06.478, x: 494.0, y: 143.0, radius: 52.6308898926
time:2017-12-03 22:24:06.674, x: 524.130432129, y: 163.670547485, radius: 72.5110168457
time:2017-12-03 22:24:06.847, x: 497.814422607, y: 145.27835083, radius: 53.8696365356
time:2017-12-03 22:24:07.023, x: 557.5, y: 172.0, radius: 47.3525009155
time:2017-12-03 22:24:07.202, x: 553.5, y: 165.0, radius: 42.7932052612
time:2017-12-03 22:24:07.379, x: 547.0, y: 165.5, radius: 54.5551185608
time:2017-12-03 22:24:07.548, x: 524.195861816, y: 163.059738159, radius: 72.2697982788
time:2017-12-03 22:24:07.729, x: 522.209350586, y: 165.209365845, radius: 71.8845825195
time:2017-12-03 22:24:07.919, x: 493.0, y: 142.0, radius: 52.801612854
time:2017-12-03 22:24:08.097, x: 493.0, y: 142.0, radius: 52.801612854
time:2017-12-03 22:24:08.267, x: 524.493164062, y: 163.012512207, radius: 72.506942749
time:2017-12-03 22:24:08.434, x: 551.952209473, y: 164.487686157, radius: 52.1465377808
time:2017-12-03 22:24:08.610, x: 523.006225586, y: 164.363830566, radius: 71.8826522827
time:2017-12-03 22:24:08.778, x: 523.447509766, y: 163.749679565, radius: 71.9914474487
time:2017-12-03 22:24:08.947, x: 521.522216797, y: 165.975265503, radius: 71.8767166138
time:2017-12-03 22:24:09.118, x: 505.5, y: 142.5, radius: 52.4453086853
time:2017-12-03 22:24:09.290, x: 525.0, y: 166.5, radius: 68.9656677246
time:2017-12-03 22:24:09.459, x: 525.0, y: 170.0, radius: 66.6034011841
time:2017-12-03 22:24:09.625, x: 522.316345215, y: 163.675109863, radius: 69.0756530762
time:2017-12-03 22:24:09.792, x: 524.5, y: 163.5, radius: 72.749671936
time:2017-12-03 22:24:09.966, x: 524.5, y: 163.5, radius: 72.749671936
time:2017-12-03 22:24:10.139, x: 523.49786377, y: 163.712097168, radius: 71.9816436768
time:2017-12-03 22:24:10.306, x: 523.49786377, y: 163.712097168, radius: 71.9816436768
time:2017-12-03 22:24:10.473, x: 523.447509766, y: 163.749679565, radius: 71.9914474487
time:2017-12-03 22:24:10.644, x: 512.669311523, y: 163.103225708, radius: 64.9388961792
time:2017-12-03 22:24:10.821, x: 494.979156494, y: 142.769348145, radius: 53.4281044006
time:2017-12-03 22:24:10.999, x: 496.73828125, y: 142.5625, radius: 52.8097686768
time:2017-12-03 22:24:11.168, x: 493.0, y: 142.0, radius: 52.801612854
time:2017-12-03 22:24:11.340, x: 500.96295166, y: 141.333328247, radius: 49.9452705383
time:2017-12-03 22:24:11.518, x: 530.524353027, y: 163.052947998, radius: 77.4160919189
time:2017-12-03 22:24:11.694, x: 517.811706543, y: 162.713943481, radius: 68.3538970947
time:2017-12-03 22:24:11.863, x: 503.0, y: 142.5, radius: 54.3531112671
time:2017-12-03 22:24:12.038, x: 503.5, y: 142.5, radius: 54.1157150269
time:2017-12-03 22:24:12.217, x: 517.022033691, y: 160.11668396, radius: 69.5520477295
time:2017-12-03 22:24:12.390, x: 516.84967041, y: 160.223495483, radius: 69.7311172485
time:2017-12-03 22:24:12.564, x: 498.998291016, y: 141.69078064, radius: 52.5570259094
time:2017-12-03 22:24:12.731, x: 491.0, y: 140.5, radius: 52.8796806335
time:2017-12-03 22:24:12.909, x: 507.272979736, y: 163.973739624, radius: 69.7240371704
time:2017-12-03 22:24:13.081, x: 492.242950439, y: 158.963546753, radius: 68.9585418701
time:2017-12-03 22:24:13.255, x: 483.708312988, y: 159.706558228, radius: 68.3609771729
time:2017-12-03 22:24:13.433, x: 470.0, y: 144.0, radius: 50.3290176392
time:2017-12-03 22:24:13.608, x: 463.17175293, y: 141.637405396, radius: 48.4780883789
time:2017-12-03 22:24:13.786, x: 467.216003418, y: 163.620742798, radius: 65.0969314575
time:2017-12-03 22:24:13.962, x: 462.532775879, y: 164.712844849, radius: 65.7210998535
time:2017-12-03 22:24:14.138, x: 439.709197998, y: 144.191940308, radius: 49.1746253967
time:2017-12-03 22:24:14.306, x: 446.5, y: 163.75, radius: 64.9640350342
time:2017-12-03 22:24:14.473, x: 441.818939209, y: 164.700271606, radius: 65.348526001
time:2017-12-03 22:24:14.655, x: 421.5, y: 143.0, radius: 52.0313415527
time:2017-12-03 22:24:14.830, x: 430.460083008, y: 165.102539062, radius: 64.2908935547
time:2017-12-03 22:24:14.997, x: 415.0, y: 149.5, radius: 55.5991096497
time:2017-12-03 22:24:15.186, x: 423.67956543, y: 167.434326172, radius: 65.8606948853
time:2017-12-03 22:24:15.365, x: 412.285064697, y: 166.122497559, radius: 63.5570220947
time:2017-12-03 22:24:15.551, x: 407.5, y: 165.0, radius: 64.6472167969
time:2017-12-03 22:24:15.733, x: 399.5, y: 165.5, radius: 64.6878128052
time:2017-12-03 22:24:15.914, x: 381.5, y: 149.5, radius: 48.8109588623
```

```
time:2017-12-03 22:24:16.091, x: 395.738677979, y: 162.268295288, radius: 59.4694480896
time:2017-12-03 22:24:16.269, x: 381.0, y: 153.0, radius: 52.4023895264
time:2017-12-03 22:24:16.445, x: 371.0, y: 148.5, radius: 52.0025024414
time:2017-12-03 22:24:16.663, x: 377.690734863, y: 168.0, radius: 65.2856826782
time:2017-12-03 22:24:16.833, x: 363.052581787, y: 165.524688721, radius: 63.5237426758
time:2017-12-03 22:24:17.007, x: 346.5, y: 148.0, radius: 48.4691619873
time:2017-12-03 22:24:17.179, x: 345.512634277, y: 168.242797852, radius: 61.7397575378
time:2017-12-03 22:24:17.360, x: 337.747344971, y: 167.673721313, radius: 61.1277198792
time:2017-12-03 22:24:17.539, x: 313.5, y: 135.0, radius: 37.8583869934
time:2017-12-03 22:24:17.717, x: 315.5, y: 151.0, radius: 45.5110855103
time:2017-12-03 22:24:17.886, x: 317.0, y: 166.5, radius: 60.6486778259
time:2017-12-03 22:24:18.051, x: 301.574279785, y: 150.700836182, radius: 45.2247619629
time:2017-12-03 22:24:18.226, x: 298.0, y: 151.5, radius: 45.4010887146
time:2017-12-03 22:24:18.397, x: 291.5, y: 147.5, radius: 46.1791000366
time:2017-12-03 22:24:18.569, x: 283.0, y: 137.5, radius: 35.5563926697
time:2017-12-03 22:24:18.748, x: 289.063659668, y: 168.373199463, radius: 61.3558387756
time:2017-12-03 22:24:18.928, x: 295.0, y: 170.0, radius: 63.5296211243
time:2017-12-03 22:24:19.107, x: 267.0, y: 153.0, radius: 47.4342651367
time:2017-12-03 22:24:19.283, x: 264.0, y: 152.0, radius: 45.4863624573
time:2017-12-03 22:24:19.456, x: 275.0, y: 170.5, radius: 60.2765464783
time:2017-12-03 22:24:19.625, x: 253.0, y: 152.0, radius: 46.0435562134
time:2017-12-03 22:24:19.792, x: 251.235992432, y: 154.764007568, radius: 44.2649421692
time:2017-12-03 22:24:19.962, x: 258.5, y: 171.676986694, radius: 59.7235107422
time:2017-12-03 22:24:20.140, x: 254.794631958, y: 171.830596924, radius: 59.6931381226
time:2017-12-03 22:24:20.324, x: 234.5, y: 154.5, radius: 49.522819519
time:2017-12-03 22:24:20.501, x: 245.199554443, y: 172.3203125, radius: 59.7954750061
time:2017-12-03 22:24:20.670, x: 218.997436523, y: 155.628204346, radius: 45.7386512756
time:2017-12-03 22:24:20.839, x: 231.033615112, y: 172.506393433, radius: 59.9406814575
time:2017-12-03 22:24:21.530, x: 211.377197266, y: 155.143280029, radius: 46.5155105591
time:2017-12-03 22:24:21.739, x: 210.0, y: 153.5, radius: 47.6052474976
time:2017-12-03 22:24:21.907, x: 171.0, y: 144.0, radius: 33.2416381836
time:2017-12-03 22:24:22.082, x: 167.0, y: 142.5, radius: 35.3023643494
time:2017-12-03 22:24:22.262, x: 149.0, y: 137.0, radius: 12.8063488007
time:2017-12-03 22:24:22.437, x: 138.5, y: 140.5, radius: 22.3272132874
time:2017-12-03 22:24:22.621, x: 139.5, y: 144.0, radius: 34.3403091431
time:2017-12-03 22:24:22.788, x: 122.5, y: 143.5, radius: 24.7892093658
time:2017-12-03 22:24:22.957, x: 128.0, y: 134.5, radius: 11.9269609451
time:2017-12-03 22:24:23.129, x: 107.0, y: 142.5, radius: 27.2810440063
time:2017-12-03 22:24:23.307, x: 100.5, y: 160.5, radius: 41.8868522644
time:2017-12-03 22:24:23.485, x: 82.5, y: 167.5, radius: 29.0259494781
time:2017-12-03 22:24:23.656, x: 98.5, y: 158.5, radius: 46.3304367065
time:2017-12-03 22:24:23.823, x: 94.5, y: 156.0, radius: 30.2697200775
time:2017-12-03 22:24:24.001, x: 65.5, y: 167.0, radius: 75.3277206421
time:2017-12-03 22:24:24.170, x: 66.0, y: 167.5, radius: 66.8301086426
time:2017-12-03 22:24:24.358, x: 58.1828842163, y: 170.919464111, radius: 67.0860671997
time:2017-12-03 22:24:24.547, x: 78.5, y: 159.5, radius: 39.3002281189
time:2017-12-03 22:24:24.738, x: 78.5, y: 157.5, radius: 43.1336135864
time:2017-12-03 22:24:24.915, x: 56.5, y: 168.899993896, radius: 66.75831604
time:2017-12-03 22:24:25.091, x: 53.0, y: 167.5, radius: 68.2807693481
time:2017-12-03 22:24:25.265, x: 67.0, y: 160.0, radius: 38.0789642334
time:2017-12-03 22:24:25.440, x: 76.8727493286, y: 160.315429688, radius: 42.8244361877
time:2017-12-03 22:24:25.626, x: 53.0444107056, y: 168.60949707, radius: 69.0574493408
time:2017-12-03 22:24:25.798, x: 78.0, y: 158.5, radius: 43.4886016846
time:2017-12-03 22:24:25.966, x: 71.0, y: 157.0, radius: 42.4382896423
time:2017-12-03 22:24:26.139, x: 64.3141403198, y: 177.20602417, radius: 73.2290344238
time:2017-12-03 22:24:26.319, x: 67.7439041138, y: 181.378051758, radius: 69.9179153442
time:2017-12-03 22:24:26.506, x: 65.7659072876, y: 182.192352295, radius: 72.1551361084
time:2017-12-03 22:24:26.714, x: 67.5, y: 158.0, radius: 38.8491668701
time:2017-12-03 22:24:26.888, x: 67.8017044067, y: 184.959564209, radius: 67.3810577393
time:2017-12-03 22:24:27.067, x: 71.0, y: 159.5, radius: 40.4012145996
time:2017-12-03 22:24:27.237, x: 74.5, y: 156.5, radius: 44.1192550659
time:2017-12-03 22:24:27.414, x: 52.5575523376, y: 168.11151123, radius: 69.0213623047
time:2017-12-03 22:24:27.598, x: 76.0, y: 157.0, radius: 44.2041702271
time:2017-12-03 22:24:27.783, x: 78.5, y: 156.5, radius: 42.3143997192
time:2017-12-03 22:24:27.958, x: 66.0, y: 206.5, radius: 68.1561737061
time:2017-12-03 22:24:28.128, x: 85.0459060669, y: 170.316650391, radius: 63.4702644348
time:2017-12-03 22:24:28.309, x: 107.5, y: 169.0, radius: 53.4439964294
time:2017-12-03 22:24:28.484, x: 107.0, y: 156.5, radius: 46.3816680908
time:2017-12-03 22:24:28.661, x: 109.5, y: 167.5, radius: 51.2299728394
time:2017-12-03 22:24:28.839, x: 100.0, y: 146.0, radius: 18.3848762512
time:2017-12-03 22:24:29.022, x: 106.0, y: 147.5, radius: 31.4842224121
```

```
time:2017-12-03 22:24:29.203, x: 124.5, y: 162.0, radius: 51.8869972229
time:2017-12-03 22:24:29.389, x: 119.5, y: 131.0, radius: 15.4030218124
time:2017-12-03 22:24:29.560, x: 112.0, y: 149.5, radius: 15.6924552917
time:2017-12-03 22:24:29.729, x: 127.0, y: 142.5, radius: 15.6924552917
time:2017-12-03 22:24:29.896, x: 134.0, y: 145.5, radius: 13.2004785538
time:2017-12-03 22:24:30.062, x: 160.792098999, y: 138.744415283, radius: 31.9476394653
time:2017-12-03 22:24:30.240, x: 168.0, y: 136.0, radius: 37.7360229492
time:2017-12-03 22:24:30.410, x: 165.0, y: 144.0, radius: 38.4188461304
time:2017-12-03 22:24:30.591, x: 178.5, y: 149.5, radius: 47.460609436
time:2017-12-03 22:24:30.765, x: 173.0, y: 140.5, radius: 29.4492073059
time:2017-12-03 22:24:30.939, x: 188.5, y: 125.0, radius: 22.8090877533
time:2017-12-03 22:24:31.112, x: 191.0, y: 148.0, radius: 45.3432350159
time:2017-12-03 22:24:31.293, x: 205.5, y: 150.5, radius: 44.6375274658
time:2017-12-03 22:24:31.469, x: 208.5, y: 138.0, radius: 34.4420776367
time:2017-12-03 22:24:31.670, x: 239.5, y: 224.5, radius: 26.6177616119
time:2017-12-03 22:24:31.842, x: 246.043228149, y: 172.162582397, radius: 62.0132980347
time:2017-12-03 22:24:32.017, x: 240.5, y: 150.0, radius: 45.4010887146
time:2017-12-03 22:24:32.188, x: 244.0, y: 149.0, radius: 41.8808822632
time:2017-12-03 22:24:32.356, x: 260.5, y: 149.5, radius: 44.7494010925
time:2017-12-03 22:24:32.528, x: 282.433227539, y: 168.960983276, radius: 62.4524726868
time:2017-12-03 22:24:32.699, x: 292.140625, y: 169.5, radius: 63.6396179199
time:2017-12-03 22:24:32.895, x: 295.801879883, y: 169.644699097, radius: 62.8400802612
time:2017-12-03 22:24:33.075, x: 291.0, y: 149.5, radius: 53.4814033508
time:2017-12-03 22:24:33.254, x: 296.5, y: 147.5, radius: 46.30884552
time:2017-12-03 22:24:33.425, x: 300.5, y: 145.0, radius: 44.9139060974
time:2017-12-03 22:24:33.598, x: 315.048187256, y: 168.294128418, radius: 62.3686981201
time:2017-12-03 22:24:33.778, x: 318.649078369, y: 168.385421753, radius: 62.5600852966
time:2017-12-03 22:24:33.962, x: 323.33026123, y: 168.906829834, radius: 62.8903427124
time:2017-12-03 22:24:34.135, x: 328.024017334, y: 167.853713989, radius: 62.7898674011
time:2017-12-03 22:24:34.307, x: 332.635314941, y: 167.992156982, radius: 62.4548110962
time:2017-12-03 22:24:34.486, x: 338.292907715, y: 168.281433105, radius: 63.4210777283
time:2017-12-03 22:24:34.672, x: 330.1902771, y: 143.123901367, radius: 46.0686149597
time:2017-12-03 22:24:34.852, x: 342.5, y: 142.5, radius: 45.6345176697
time:2017-12-03 22:24:35.037, x: 353.0, y: 165.5, radius: 62.6997795105
time:2017-12-03 22:24:35.214, x: 349.753509521, y: 146.809860229, radius: 49.7460670471
time:2017-12-03 22:24:35.391, x: 370.5, y: 166.719299316, radius: 63.6256141663
time:2017-12-03 22:24:35.563, x: 372.0, y: 140.0, radius: 44.2832794189
time:2017-12-03 22:24:35.731, x: 382.63873291, y: 144.106933594, radius: 45.6851501465
time:2017-12-03 22:24:35.911, x: 396.979888916, y: 165.290466309, radius: 66.6301651001
time:2017-12-03 22:24:36.090, x: 405.780517578, y: 165.752578735, radius: 66.1322097778
time:2017-12-03 22:24:36.267, x: 411.967102051, y: 167.0, radius: 64.0518951416
time:2017-12-03 22:24:36.443, x: 423.178405762, y: 166.045578003, radius: 64.9613113403
time:2017-12-03 22:24:36.651, x: 433.493499756, y: 165.363632202, radius: 67.4157409668
time:2017-12-03 22:24:36.819, x: 440.069122314, y: 165.772842407, radius: 63.741558075
time:2017-12-03 22:24:37.006, x: 454.297363281, y: 165.223937988, radius: 71.0793991089
time:2017-12-03 22:24:37.180, x: 452.5574646, y: 163.18347168, radius: 66.72290802
time:2017-12-03 22:24:37.365, x: 463.219116211, y: 163.843948364, radius: 67.4571685791
time:2017-12-03 22:24:37.543, x: 465.310577393, y: 147.486785889, radius: 53.225479126
time:2017-12-03 22:24:37.721, x: 477.064208984, y: 161.687789917, radius: 65.8137283325
time:2017-12-03 22:24:37.901, x: 497.672424316, y: 159.913787842, radius: 64.0602340698
time:2017-12-03 22:24:38.070, x: 504.672119141, y: 162.491882324, radius: 68.6128082275
time:2017-12-03 22:24:38.239, x: 513.841491699, y: 162.867126465, radius: 68.885345459
time:2017-12-03 22:24:38.417, x: 515.523803711, y: 137.416351318, radius: 49.3813209534
time:2017-12-03 22:24:38.593, x: 515.49786377, y: 141.251052856, radius: 55.7500724792
time:2017-12-03 22:24:38.766, x: 523.0, y: 157.0, radius: 67.9559707642
time:2017-12-03 22:24:38.938, x: 535.0, y: 159.263153076, radius: 71.4991989136
time:2017-12-03 22:24:39.121, x: 536.606689453, y: 159.899215698, radius: 70.7592926025
time:2017-12-03 22:24:39.295, x: 540.0, y: 161.528930664, radius: 71.1141967773
time:2017-12-03 22:24:39.468, x: 538.529418945, y: 138.361343384, radius: 50.6016731262
time:2017-12-03 22:24:39.648, x: 539.295654297, y: 159.285995483, radius: 70.3970413208
time:2017-12-03 22:24:39.829, x: 536.0, y: 160.214874268, radius: 70.1274108887
time:2017-12-03 22:24:40.010, x: 528.5, y: 158.785720825, radius: 67.3118209839
time:2017-12-03 22:24:40.189, x: 533.5, y: 162.394729614, radius: 70.8192520142
time:2017-12-03 22:24:40.368, x: 532.0625, y: 157.9375, radius: 66.0578994751
time:2017-12-03 22:24:40.538, x: 523.539367676, y: 163.126831055, radius: 66.9609451294
time:2017-12-03 22:24:40.712, x: 526.285705566, y: 162.714279175, radius: 70.7341461182
time:2017-12-03 22:24:40.894, x: 510.0, y: 142.5, radius: 54.3531112671
time:2017-12-03 22:24:41.069, x: 510.873291016, y: 136.063354492, radius: 49.7892341614
time:2017-12-03 22:24:41.246, x: 531.966186523, y: 172.42930603, radius: 66.2950744629
time:2017-12-03 22:24:41.421, x: 528.0, y: 164.0, radius: 72.8629837036
time:2017-12-03 22:24:41.638, x: 511.718261719, y: 137.338470459, radius: 49.8334655762
```

```
time:2017-12-03 22:24:41.807, x: 526.13671875, y: 163.374542236, radius: 71.0832061768
time:2017-12-03 22:24:41.974, x: 526.010009766, y: 163.5, radius: 71.0794143677
time:2017-12-03 22:24:42.157, x: 510.759674072, y: 133.660217285, radius: 50.2213363647
time:2017-12-03 22:24:42.325, x: 529.573059082, y: 173.289978027, radius: 64.1467208862
time:2017-12-03 22:24:42.492, x: 526.889648438, y: 162.629074097, radius: 71.1149291992
time:2017-12-03 22:24:42.661, x: 528.815002441, y: 163.662918091, radius: 73.2289657593
time:2017-12-03 22:24:42.836, x: 529.776489258, y: 162.116928101, radius: 72.9089508057
time:2017-12-03 22:24:43.007, x: 528.0, y: 164.0, radius: 72.8629837036
time:2017-12-03 22:24:43.179, x: 528.854187012, y: 162.077835083, radius: 72.1926193237
time:2017-12-03 22:24:43.363, x: 512.261535645, y: 138.158355713, radius: 49.8866882324
time:2017-12-03 22:24:43.530, x: 533.0, y: 173.0, radius: 67.00756073
time:2017-12-03 22:24:43.705, x: 526.5, y: 164.0, radius: 71.7792739868
time:2017-12-03 22:24:43.892, x: 525.062561035, y: 164.438049316, radius: 71.0652770996
time:2017-12-03 22:24:44.079, x: 526.5, y: 164.0, radius: 71.7792739868
time:2017-12-03 22:24:44.258, x: 530.5, y: 171.5, radius: 66.0038833618
time:2017-12-03 22:24:44.425, x: 513.0, y: 141.5, radius: 49.8724365234
time:2017-12-03 22:24:44.593, x: 526.782043457, y: 162.218688965, radius: 70.7562103271
time:2017-12-03 22:24:44.759, x: 526.515014648, y: 163.0, radius: 71.0971755981
time:2017-12-03 22:24:44.930, x: 526.889648438, y: 162.629074097, radius: 71.1149291992
time:2017-12-03 22:24:45.098, x: 528.0, y: 164.0, radius: 72.8629837036
time:2017-12-03 22:24:45.279, x: 528.0, y: 164.0, radius: 72.8629837036
time:2017-12-03 22:24:45.460, x: 526.02722168, y: 163.482925415, radius: 71.0799179077
time:2017-12-03 22:24:45.636, x: 512.332580566, y: 139.083709717, radius: 49.494556427
time:2017-12-03 22:24:45.811, x: 505.0, y: 132.5, radius: 47.9505958557
time:2017-12-03 22:24:45.987, x: 526.5, y: 164.0, radius: 71.7792739868
time:2017-12-03 22:24:46.154, x: 511.875793457, y: 135.790634155, radius: 50.7808265686
time:2017-12-03 22:24:46.331, x: 512.395446777, y: 137.098312378, radius: 49.9544906616
time:2017-12-03 22:24:46.505, x: 526.543640137, y: 163.955032349, radius: 71.7793045044
time:2017-12-03 22:24:46.698, x: 526.010009766, y: 163.5, radius: 71.0794143677
time:2017-12-03 22:24:46.866, x: 534.0, y: 175.0, radius: 66.6109085083
time:2017-12-03 22:24:47.044, x: 526.010009766, y: 163.5, radius: 71.0794143677
time:2017-12-03 22:24:47.219, x: 510.5, y: 134.0, radius: 50.6286468506
time:2017-12-03 22:24:47.402, x: 533.0, y: 173.0, radius: 66.2194290161
time:2017-12-03 22:24:47.577, x: 526.13671875, y: 163.374542236, radius: 71.0832061768
time:2017-12-03 22:24:47.756, x: 526.0, y: 163.0, radius: 70.7249145508
time:2017-12-03 22:24:47.932, x: 511.121002197, y: 134.189498901, radius: 49.393825531
time:2017-12-03 22:24:48.116, x: 510.5, y: 134.0, radius: 50.6286468506
time:2017-12-03 22:24:48.317, x: 527.299255371, y: 162.651031494, radius: 70.7519760132
time:2017-12-03 22:24:48.497, x: 534.6875, y: 175.5859375, radius: 66.4024276733
time:2017-12-03 22:24:48.671, x: 528.624206543, y: 162.436233521, radius: 72.2609786987
time:2017-12-03 22:24:48.840, x: 527.136108398, y: 162.480911255, radius: 71.1940536499
time:2017-12-03 22:24:49.021, x: 509.5, y: 133.5, radius: 50.6212425232
time:2017-12-03 22:24:49.204, x: 512.205688477, y: 138.074035645, radius: 49.8803062439
time:2017-12-03 22:24:49.388, x: 511.0, y: 134.5, radius: 50.7568702698
time:2017-12-03 22:24:49.569, x: 527.120483398, y: 162.481384277, radius: 71.1828460693
time:2017-12-03 22:24:49.748, x: 532.5, y: 175.0, radius: 65.4007873535
time:2017-12-03 22:24:49.919, x: 527.155151367, y: 162.844833374, radius: 70.7764511108
time:2017-12-03 22:24:50.093, x: 506.0, y: 133.0, radius: 46.8615989685
```