ROBOGRAPHERS

FACIAL EXPRESSION RECOGNITION USING SWARMS

Final Report MRSD project

SPONSOR: DR. KATIA SYCARA

May 5, 2016

TEAM G

JIMIT GANDHI

GAURI GANDHI

ROHIT DASHRATHI

SIDA WANG

TIFFANY MAY

ABSTRACT

This is a comprehensive report that reviews the design and development of a system of swarm mobile robots which would collaboratively analyze facial expressions and click pictures in a social occasion such as parties, weddings, pleasant ceremonies. The development of this project required us to enquire and discuss various requirements that would be critical to any usual social occasion use case. Once requirements were defined, a series of discussions and brainstorming lead to designing of functional and cyber-physical architecture involving various subsystems. This in turn led to development of a project schedule which lays down dates for completion of step-by-step progress and milestones thus fulfilling the requirements slowly. This report describes each of the above aspects in detail.

The preliminary execution of this schedule was carried out and results were recorded. Analyzing these results gave us a sufficient proof of the feasibility of the planning stage as well as brought to light various potential risks and problems, some of which were already taken care of. The entire report concludes with the lessons and takeaways from the experience till date and in addition also lists down the future work that can be done.

Table of Contents

1. Project Description

If photographers had robotic assistants capturing happy moments in events such as weddings, birthdays or graduation ceremonies? 'Robographers' is the preliminary effort aimed at developing such autonomous assistants that not only click photos, but also recognize and capture the human expressions accurately with equal competency. The principle of the project is facial expression recognition and accurate head pose tracking using a swarm of robots. Instead of working individually, a swarm of mobile robots will work collaboratively to accurately estimate the human expressions and click snaps accordingly.

By using multiple cameras, it is possible to improve the estimate of the facial expressions and head pose through redundant noisy measurements and better handling of real-world issues such as occlusions. By adding mobility to the cameras, thus making each camera a dynamically actuated information source, it is possible to further improve the estimate while improving the tracking of the human as they move through an uncontrolled environment.

The project Robographers has been conceptualized by the common vision of a group of students who believe that robots can perform better when they work in collaboration with each other and also with the humans while at the same time ensuring the safety and comfort of the humans in the environment. The goals of the project are as follows

1. The first component of the project involves designing and building the accurate pan-tilt units for the cameras and incorporating multi-camera head tracking into the developed sensor fusion software.

2. The subsequent component involves the development of software packages for the detection of a person of interest and navigation of the 3-robot swarm towards the same. The flocking of 3 robot swarm was carried out using the robot mobility basics such as coordinate frame transformations where the hardware support provided in the form of global cameras.

3. The third component involves evaluating multiple cameras and applying computer vision and sensor fusion techniques to select the best set of the information from multiple static cameras using ROS and IntraFace[1] to obtain the most accurate facial pose and expression estimate. IntraFace is a facial expression recognition software developed by the Human Sensing Lab at the Carnegie Mellon University.

4. The final component involves the development and implementation of the software package development for successful navigation of the Turtlebot swarm in front of the person of interest along with the smiling face detection using the IntraFace software. The best smiling pose is recorded by the static camera in the best position to detect the face and the picture is captured.

2. Use Case

“Rohit Dashrathi, the luckiest man in the world, is going to have a big wedding today. And he has got a thought to introduce some new elements to this joyful ceremony.

One of his friends introduced him to the latest product from the Gandhi Inc., a group of three robots which can take candid pictures of happy and smiling people collaboratively. This service can be rented from the company and is charged on an hourly basis.

Hosted in a 216 square feet area of the Sida hall and attended by 50 guests, this wedding ceremony is going to last more than three hours. The situation is mostly out of control for professional human photographers. But Rohit is not worried about it, as he has delegated the photography of this very special day in his life to a very special army of robotic photographers, known as the “Robographers”.

Throughout the event, Robographers have become a hot topic of discussion. These amusing robots wearing lovely dresses roam around to add a great charm to the wedding. Adjustable pan-tilt cameras on these units can capture photos of people of all heights, leaving no one unattended. The collision avoidance system keeps all the guests safe. Based on the processing speed greater than 20 Hz, Robographers can detected the human, recognize happy faces and take pictures at the very moment when they smile with joy. The most surprising part is that they are able to take candid pictures without drawing too much attention of the subjects and all of the pictures are very natural and of high quality.

Figure 1: The Robographers in action (use case)

Besides recognizing smiling faces, these robots also remember people's faces and thus avoid taking pictures of the same guest too many times. Armed with the head pose tracking technology, the Robographers can tell if the picture is good enough or not by the angle of the faces in the image. The best part is that they can communicate with each other and find all the smiling faces together.

Eventually the robots cover the whole event on both time scale and space scale, which is a

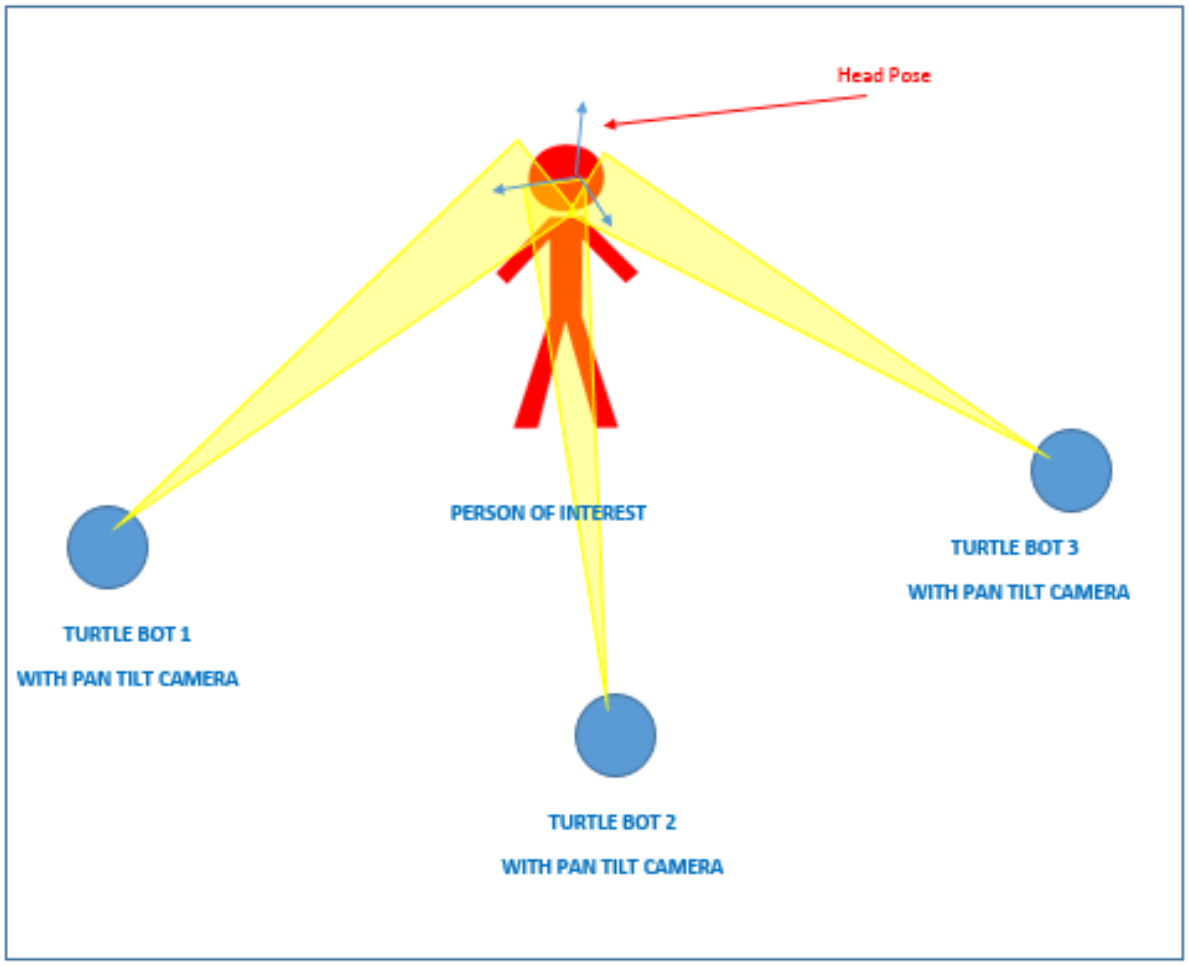

Figure 2: Turtlebot detecting the person of interest.

high standard where no human photographer can reach. All the guests get natural and smiling pictures in their e-mail boxes sent directly from the Robographers. To Rohit's surprise, the cost of hiring these autonomous photographers is half but the number and quality of pictures is doubled. This is the best wedding ever."

-Article published in Tiffany Times, May 2015

3. System Level Requirements

3.1 Mandatory System Level Requirements

3.1.1 Functional Requirements

Robots in the system shall:

M.F.1: Detect Humans using April Tags in <**1s**

M.F.2: Recognise **Smiling** Expression in <**0.4s**

M.F.3: Flock Autonomously Between Multiple Locations at **15-20cm/s**

M.F.4: Click Photos In **<1.5 S** Response Time after Expression Detection

M.F.5: Click Photos When Individual/Collaborative Smile and Head Pose Assessment **>50%**

M.F.6: Take Pictures within **3-4 Ft** Range

M.F.7: Click At Least **70 %** Smiling Photos (Measure of Overall Performance Requirement)

3.1.2 Non-Functional Requirements

Robots in in the system shall:

M.N.1: Be Supported With Good Lighting Conditions (Fully Illuminated Face All the Time)

M.N.2: Have Wireless Communication mode

M.N.3: Have adjustable elevation

M.N.5: Be easy to operate

M.N.6: Should maintain physical stability (Robots should not topple)

M.N.7: Weigh not more than 11 kg

M.N.8: Should have minimum 3 robots in SWARM

3.2 Desirable System Level Requirements

3.2.1 Functional Requirements

Robots in the system should:

D.F.1: Should identify multiple expressions such as Sad, Disgust, Happy, Scared, Surprise, Neutral, blinks

D.F.2: Should detect multiple faces

D.F.3: Search for human figures quickly

D.F.4: Navigate as quickly as possible

D.F.5: Be able to print photos

D.F.6: Work throughout an event (Avg. 4 Hrs.)

D.F.7: Drive autonomously faster between multiple locations At 40 cm/s

D.F.8: Detect Obstacles on the surface

D.F.9: Take Pictures within 20 Ft Range

3.2.1 Non-Functional Requirements

Robots in the system should:

D.N.1: Have flash light for good lighting to capture photos

D.N.2: Have a Graphical User Interface

D.N.3: Not find same person again and again

D.N.4: Have small setup time

D.N.5: Have automatic adjustable elevation

D.N.6: Incorporate 6 robots in swarm

D.N.7: Have minimum 2 Hrs. of battery time

4. Functional Architecture

The functional architecture for Robographers is as shown in figure 2, which can be divided into 3 subsystems: Human detection, planning-navigation & Face-smile detection. It is structured such that the inputs & outputs are on the left and its internal architecture is enclosed in 3 boxes, one for each subsystem on the right hand side.

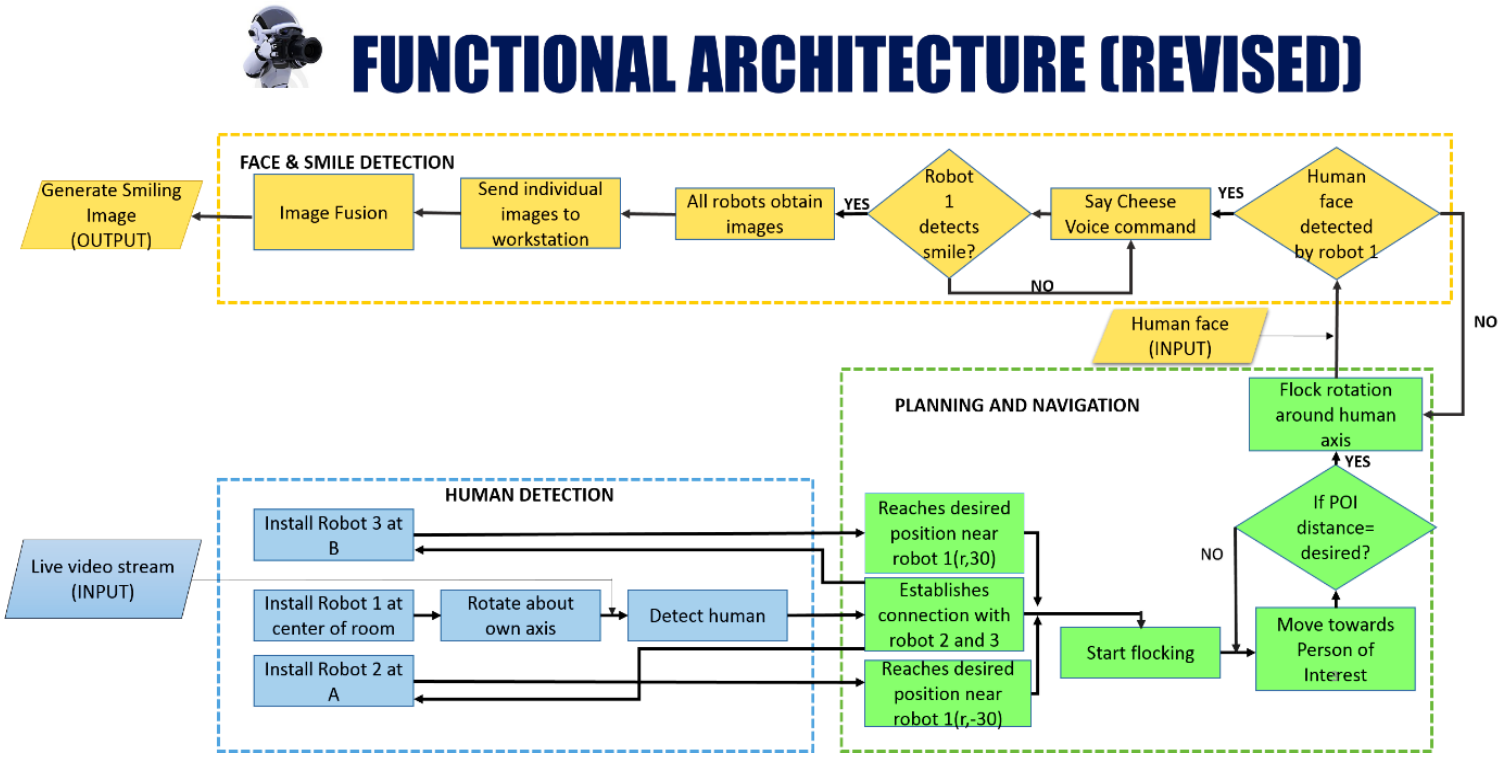

Figure 3: Functional Architecture

Work flow:

1. Turtlebots 1, 2 & 3 will be installed prefixed locations in the specified room. Location for the Turtlebot 1 will be the center of the room while the locations for the robots 2 & 3 will be the any 2 corners of the room.
2. Robot 1 at the center of the room will rotate about its own axis.
3. Web camera on the robot 1 will obtain live video stream (input to the system) of the surroundings in the room while rotating.
4. The camera will look for the persons in the room and will detect the very first person seen through its video feed.
5. Once the desired person is detected, the robot will stop rotating and stay stationary at the position at which it detected the first person. This completes the working of the human detection subsystem. The detected human will be an input for the working of the subsequent planning and navigation subsystem.
6. After human detection, the robot 1 will communicate with the other two robots. Robots 2 & 3 will perform relative localization and will look for the position of the robot 1 using the cameras mounted on them. After detecting the robot 1 position, they will

navigate towards robot 1 at their prefixed relative positions with respect to robot 1.

7. All 3 robots will form a flock and will start moving collaboratively towards the person of interest, detected by the robot 1, performing relative localization.
8. The flock will start moving when the flock reaches a position which is equal to a prefixed desired distance from the person of interest (1 meter). Note that these robots will be oriented in the same manner as decided in step 6. This means that each of the 3 robots will be at 1 meter distance from the human, at angles -30 degrees (robot 2 position, 0 degrees (robot 1 position), 30 degrees (robot 3 position) respectively. Robot 1 will at the 0 degrees position, which means that it will stand at 1 meter from the person of interest, where the person will be facing the camera directly.
9. If the person turns his/her head, the flock will rotate around the human, with the camera on the Turtlebot 1 trying to find the persons face. The system will reorient itself around human at -30 degrees, 0 degrees, 30 degrees respectively as explained before in step 8.
10. After detecting the face, the human will hear a voice request from the system, saying 'say cheese', asking him to smile. Face detection will be performed by the pan tilt camera unit motor movements.
11. The cameras on all the 3 robots detect the smiles and compare the smiling estimates with each other. The camera with the best smile estimate will click the photo.
12. Images clicked by the robot will be sent back to the workstation. Workstation will display and store the image.
13. The final output will be the generation of an accurate and clear smiling image.

5. System Level Trade Studies

1. Camera

Low cost camera for image capture												
Parameter	Level	Weight Factor	**Logitech HD pro c920**	Logitech HD c615	Genius widecam F100	HP HD 4310	Brother NW-1000	Gear Head WC8500HD	Hue HD Camera	Digital Innovations ChatCam HD 1080P	Creative Live! Cam Chat HD	Freetalk Everyman HD Webcam
Price (Lesser the better)	4	0.75	**4**	3	6	5	1	9	6	5	9	5
ZOOM (More the better)	4	1	**10**	0	0	0	7.5	7.5	0	10	10	0
Compatibility with linux	1	1.75	**10**	10	10	10	0	10	10	0	10	10
Video Speed	4	1	**10**	5	10	10	10	7.5	7.5	10	7.5	2.5
Resolution pixel (More the better)	3	1.25	**10**	6	8	8	6	6	2	2	4	2
Autofocus	3	1.25	**10**	10	5	10	10	10	5	10	0	10
Face tracking by itself	2	1.5	**10**	0	0	0	0	0	0	0	10	0
Face recognition by itself	2	1.5	**10**	0	0	0	0	0	0	0	0	0
TOTAL		10	**9.55**	4.475	4.825	5.37	3.825	5.925	3.825	3.875	6.175	3.875

Table 1 Camera Trade Study; Selected Camera: Logitech HD Pro c920

Criterions:													
Wt factor		**Price**		**Range**		**Zoom**		**Focus**		**Resolution**		**Speed**	
1	1.75	0-10	10	0-3 Ft	2	0x	0	Auto	10			30fps at 1080p	10
2	1.5	0-20	9	3-6 Ft	4	1x	2.5	Manual	5	>=2.1	2	30fps at 720p	7.5
3	1.25	21-30	8	6-9 Ft	6	2x	5	Fixed	0	5.7	4	30fps at 480p	5
4	1	31-40	7	9-12 Ft	8	3x	7.5			7.5-8	6	22fps at 720p	2.5
5	0.75	41-50	6	12-15 Ft	10	4x	10			12-13.	8		
		51-60	5							15	10		
		61-70	4										
		71-80	3										
		81-90	2										
		91-100	1										

Table 2 Camera Criterion Table

2. Chromebooks for Turtlebots

Trade Study: Chrome-books														
Parameters	Asus netbook Eee_PC_1025Ce	Asus Chromebook C201		Chromebook pixel		hp chromebook 11 G4	hp chromebook 14 G3	Toshiba chromebook 2	Acer Chromebook 11	Acer Chromebook 11 (C720-3404)	Acer Chromebook 13	Asus Chromebook C300	Asus Chromebook C200	
Price (Lesser the better)	5	8	9	0	0	8	8	6	9	5	6	7	6	5
Weight	10	10	10	7.5	7.5	10	2.5	7.5	10	10	7.5	7.5	10	10
Memory	8.75	1.25	1.25	2.5	3.75	1.25	1.25	1.25	1.25	2.5	1.25	1.25	1.25	2.5
RAM	5	5	5	10	10	7.5	5	7.5	5	7.5	7.5	7.5	5	7.5
Ubuntu compatibility (proc	10	0	0	10	10	10	0	10	10	10	10	10	10	10
Battery life	8.75	10	10	8.75	8.75	2	1.25	3.75	3.75	2.5	3.75	5	7.5	7.5
Display size	10	8	8	6	6	8	5	6	8	8	6	6	8	8
Processor spec	2	2	2	8	8	2	2	2	2	6	2	2	2	2
TOTAL	8.02	5.45	5.62	7.33	7.51	6.92	3.52	6.35	6.87	7.21	6.35	6.66	6.78	7.208

Table 3 Chromebook Trade Study; Selected Chromebook: Acer Chromebook C720

Criterions:

Wt factor			Price		Weight		Speed		Memory	
1	1.75	1.75	100-150	10	1kg<X<1.5 kg	10	<1.5Gz	1.25	16GB	1.25
2	1.5	1.5	150-200	9	1.5kg<X<2 kg	7.5	1.5Ghz	2.5	32GB	2.5
3	1.25	1.25	200-250	8	2kg<X<2.5 kg	5	1.6Ghz	3.75	32<X<100GB	3.75
4	1	1	250-300	7	2.5kg<X<3 kg	2.5	1.7Ghz	5	100<X<200	5
5	0.75	0.75	300-350	6			1.8Ghz	6.25	200<X<300	6.25
			350-400	5			2.16Gz	7.5	300<X<400	7.5
			400-450	4			2.2Ghz	8.75	400<X500	8.75
			451-500	3			>2.2Ghz	10	>500	10
			501-550	2						
			551-600	1						
			600>	0						
Battery Life		**Display size**		**Processor**		**RAM**		**Ubuntu compatibility**		
6	1.25	10.1	10	Celeron	2.5	1GB	2.5	Intel	10	
8	2	11.6	8	Dual core/i3	5	2 GB	5	Non-intel	0	
8.5	2.5	13.3	6	Quad core/i5	7.5	4GB	7.5			
9	3.75	15	4	Octa core/i7	10	8GB	10			
9.5	4.5	>15	2							
10	5									
10.5	6.25									
11	7.5									
12	8.75									
13	10									

Table 4 Chromebook Criterion Table

6. Cyber-Physical Architecture

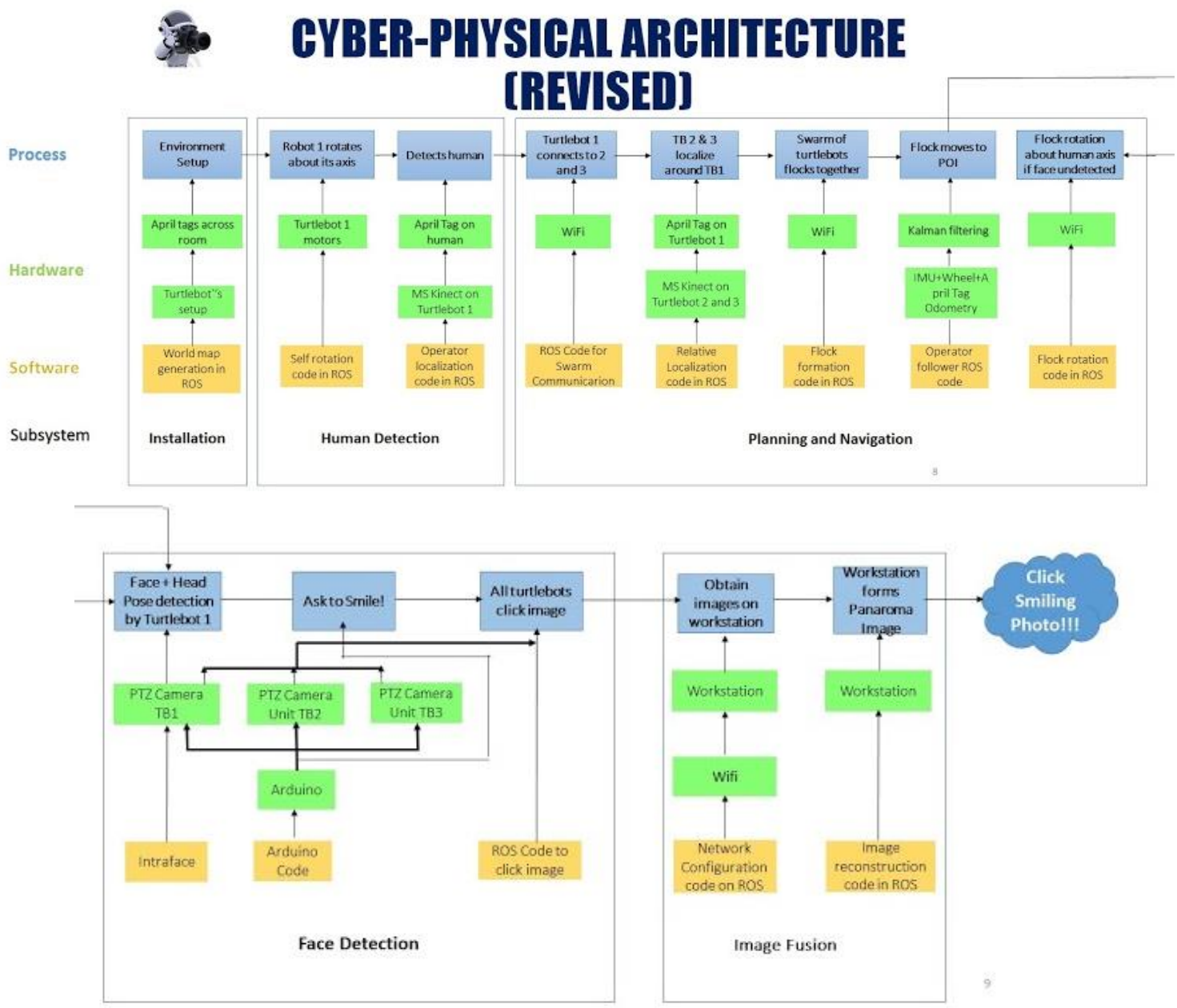

Figure 4: Cyber-Physical Architecture

The cyber physical architecture in complete synchronization with the functional architecture. To make it easier to get the whole idea, the cyber-physical architecture is divided into the same 3 subsystems Human detection, planning-navigation & Face-smile detection. The design of the cyber physical architecture is such that it extends the functional architecture to give a realization of methods/tools used for getting done the functions involved. It depicts the software and hardware components used in each subsystem to complete the tasks expected.

Now we will look at these parts one by one. The installation box is not technically a sub system. It just explains the basic installation procedure before putting the system to work. When the Turtlebots are installed in desired fashion in the specified room, the robot 1 initiates the self -rotation to find the person of interest. All the persons in the rooms are required to carry

an April Tag on their shirt for the experiment. The self-rotation of Turtlebot 1 will be carried out using a self -rotation node written in ROS, which will be fed to the Turtlebot motors. The pan tilt unit cameras mounted on the Turtlebot 1 will then try to detect the first person with April Tags on his chest. This will be carried out using the human localization code written in ROS. When the human is detected by the pan tilt camera unit on the Turtlebot 1, the robot will discontinue rotation and will stop at its place. It will then communicate with other 2 Turtlebots over Wi-Fi. This communication will initiate the working of the Turtlebots 2 & 3. They will then localize the linear and angular position of the Turtlebot 1. This will be achieved by the detection of April Tags mounted on the Turtlebot 1, by the pan tilt camera units mounted on the Turtlebots 2 & 3. These 2 Turtlebots, in this way, will estimate the position of Turtlebot 1 and will navigate toward it. They will stop after reaching the desired relative position with respect to the Turtlebot 1. The 3 robots will form a flock in this way, as a result of the flock formation node written in ROS. The flock, as a system, will then start navigating towards the detected person of interest. It will stop navigating once it reaches the desired '1 meter away from human' position. This completes the working of the human detection followed by the working of the planning-navigation subsystem.

When the system reaches desired position with respect to human, the pan tilt unit on the robot 1 will try to detect the face of the human. This will be carried out using the face tracking algorithm written in Arduino IDE format which will be fed to the pan tilt motors through the Arduino Mega microcontroller. If the face is not detected, the system will conclude that the face of the person is turned in some other direction. After this observation, the flock system will perform a rotation around the human, considering human as an axis of rotation. During this rotation, the pan tilt camera unit mounted on the Turtlebot 1 will keep performing its motions to detect the face of the person. The system will stop rotating once it detects the person's face.

After the face detection, the system will try to detect the expressions of the person. The expression detection part will be carried out by the IntraFace software, active on all the computers mounted on the 3 Turtlebots. The IntraFace will try to find the expression of the person of interest. When it recognises that the person is smiling, it will try to find the smile estimate as a percentage value. This is an inbuilt facility offered by IntraFace. The respective smile percentages estimated by all the 3 cameras on each Turtlebot will be compared to each other to find the maximum amongst the set. The Turtlebot with the maximum smile percentage will be issued a command to click the photo. This will be done by the combination for the face tracking Arduino code, the face & expression detection IntraFace facility and the photo capture ROS node. The clicked photos will then be transferred wirelessly (using Wi-Fi) to the workstation laptop, not mounted on the system. This will yield an accurate and clear smiling photo of the person, which the output of this system.

7. System Description and Evaluation

7.1 Subsystem Description

The Robographers system has the following subsystems (refer functional architecture) as follows:

7.1.1 Mechanical Subsystem

The mechanical subsystem consists of the pan-tilt elevation axis. The pan-tilt-elevation rod are to be mounted on the three Turtlebots. The procedure followed were Initial Design, Finite Element Analysis, Prototyping, Design changes followed by final fabrication.

Initial Design

While designing the pan-tilt unit + elevation rod system, following things were taken into account.

- Weight of the pan-tilt +elevation rod > payload capacity of Turtlebot. (7lbs)
- Design should be stable such that there should be vibrations on the camera if the Turtlebot is moving
- The elevation rod should be flexible in the sense that we can set the height of the camera according to our will.
- Design should be easy to mount and unmount.

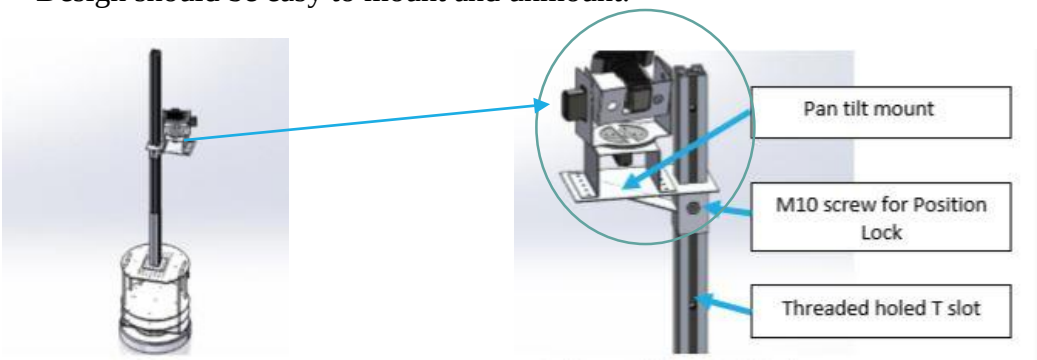

Figure 5: Initial design of Pan-tilt +Elevation rod unit mounted on turtle bot

Finite Element Analysis

After the design part, analysis was done to cross check if the above requirements were met. The first design introduced tremendous vibrations. Using CAD animation tool box and after detailed analysis of the structure, it was concluded that the base of the elevation rod had four critical points known as “failure points”. Once these points were identified, a new design was proposed. On performing similar testing and analysis the vibrations were minimized to a great extent and thus was now within acceptable standards.

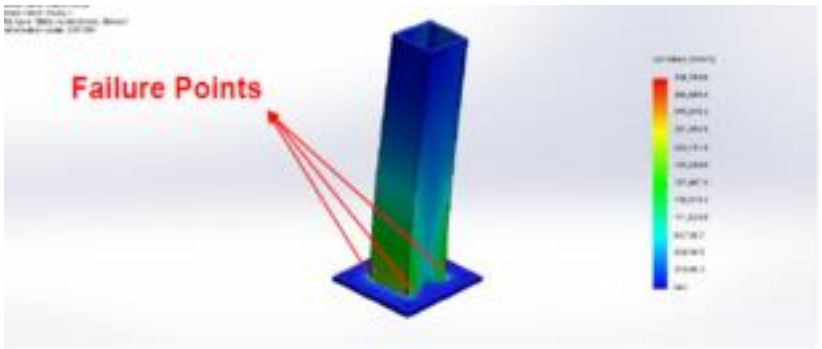

Figure 6: Failure points as identified in the initial design at the base of the elevation rod which connects the rod to the Turtlebot.

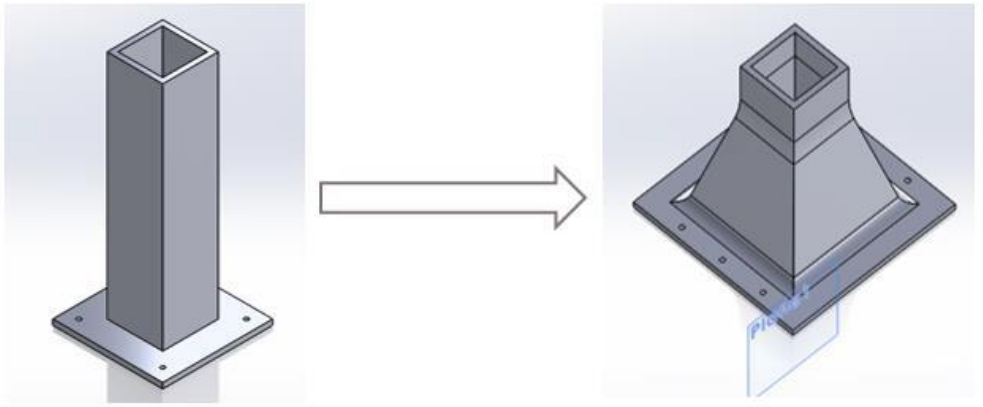

Figure 7: Change in design from initial design which had 4 failure points. The new design is the final workable design.

Prototyping:

Finally, the elevation rod holder bracket was manufactured using the additive manufacturing technique provisioned by the Makerbot 2X replicator machine in the lab. After manufacturing the brackets, they were again assembled with the 4cm x 4cm 2.5 Ft Aluminium extrusion bar and the complete assembly was finally mounted on the Turtlebot. Figure 8 shows the complete assembly with the Pan Tilt unit prototype.

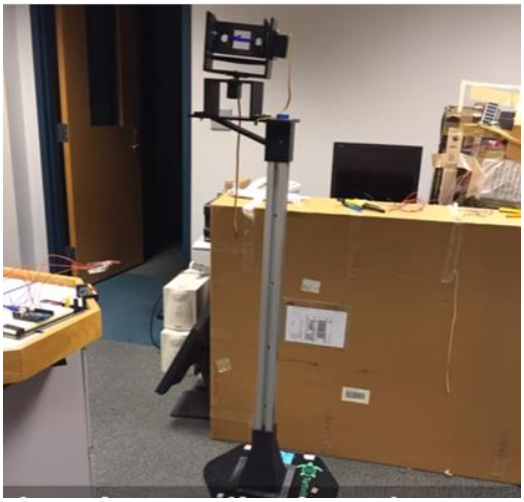

Figure 8: Protoype-1 of the Pan tilt elevate unit

Design changes:

The prototype 1 as shown in the above figure was tested and following issues were found out:

a) Mechanical restriction over the pan/tilt motions:

The Tilt servo in the existing model is mounted on the outer side of the pan tilt unit. Hence, the motion of the panning servo was often obstructed by the elevation rod, causing the incomplete utilization of the servo range. This was considered while redesigning the pan tilt unit and was fixed by allowing the motor mounting on the inner side of the camera mounting bracket. Similarly, as the existing pan tilt unit is fabricated using additive manufacturing, a reinforcement plate was added between the tilt brackets to avoid the bending in its edges due to any load. However, this reinforcement plate acted as an obstruction for the tilt servo motion.

Hence, it was not possible to track the human beyond a certain range in the vertical axis. This issue has been solved in the new design of the pan tilt unit as there is no external mechanical restriction to the tilt motor.

b) Bigger overall dimensions and high self-weight:

The current pan tilt unit has somewhat huge dimensions (18.5 cm height, 9.525 cm width, and 14 cm length). Also, it weighs around 500 gms which is 10% of the Turtlebot payload capacity of 5kgs. As the other components mounted on it such as the servo motors and the camera have very small sizes with low weights, it is unnecessary to have such a bulky structure. Besides, it is always better to have compact things regardless of any of the flaws in its predecessor. This has been taken care of in the new design of the pan tilt unit. The overall dimensions of the new pan tilt unit are very small as compared to the existing on (13.7cm height, 7.8 cm length, 7 cm width). The height is reduced by 26%, length by 44% while the breadth is reduced by 27%. In addition, its weight is 300 gms, which is just 60% of the existing pan tilt unit.

3) High system inertia:

The existing pan tilt elevation mechanism uses a 4 cm X 4 cm cross section, 2'8" Ft long 80x80 Aluminium extrusion bar for Elevation. This bar itself weighs 1.6 kgs which imposes high inertia on the Turtlebot. This high inertia results in the high amplitude vibrations when the system starts/stops. These vibrations make the system unstable and also affect the quality of human detection, expression detection and the photo capture. Also, it was noted during FVE and subsequent testing session that the height of the elevation bar was often under-utilized. Hence, the new design uses a 3cm x 3cm, 2'5" Ft long 80x80 Al extrusion bar, weighing just 600 gms. This reduction in the overall dimensions by 25% and the reduction in weight by 60% of the elevation bar is supposed to lower the effect of inertia and subsequently the vibrations. Based on these observations, design changes were made in initial pan tilt unit as shown in the figure 9:

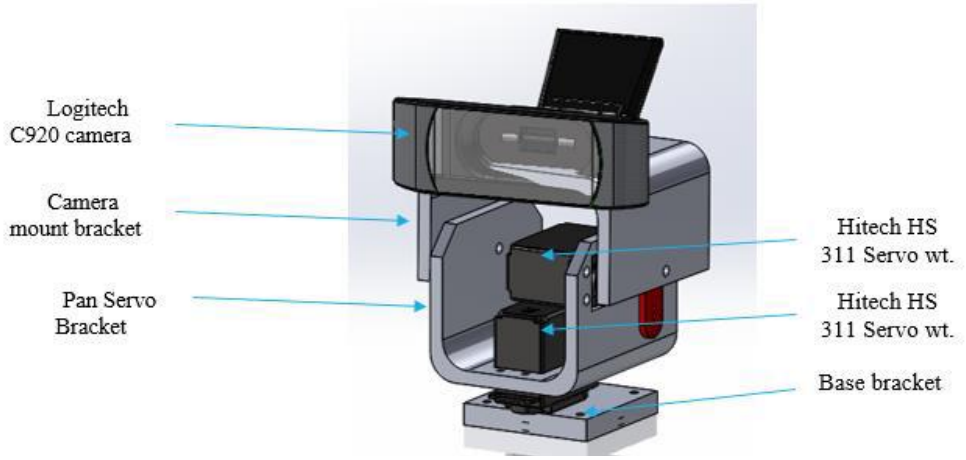

Figure 9: Pan Tilt elevate unit- revision 2

Final fabrication:

Figure 10 shows the redesigned pan tilt unit which depicts the following design achievements:

1. Full utilization of the pan and tilt servo limits
2. 26%, 44%, 27% reductions in the height, length and width of the pan tilt unit respectively.
3. 60% reduction in the pan tilt elevate assembly weight
4. Better design to overcome inherent vibrations due to inertia.

Figure 10: Final pan tilt unit

Completion of the mechanical assemblies:

After completion of the pan tilt unit assemblies, next task was to integrate these assemblies with their respective Turtlebots. Before the PR#10, I had already assembled the 2 remodelled pan tilt units (rev-2) and had integrated them with their respective elevation rods and the Turtlebots. Before the PR#11, I decided to integrate the 3rd pan tilt unit along with the third Turtlebot. This also required using new elevation rod of 30cm x 30cm size along with the newly designed pan tilt holder and elevation rod holder parts. However, the latter 2 parts take around 8 hours each for their fabrication using 3D printing. Hence, to save on time, I decided to use the same Turtlebot, already fitted with the previous 40cm x 40 cm elevation rod along with the pan tilt holder (rev-1) and elevation rod holder (rev-1) shown in the figure 10. Figure 11 represents the complete Robographers system consisting of 3 Turtlebots integrated with the pan tilt elevate units.

Figure 11: Final Mechanical Assembly

7.1.2 Swarm Subsystem

Turtlebot packages are designed such that each package treats a Turtlebot as a separate master. One cannot have separate Turtlebots over same master. For this, ROS developers have introduced a multi-master framework called ROCON which stands for Robotics in Concert.

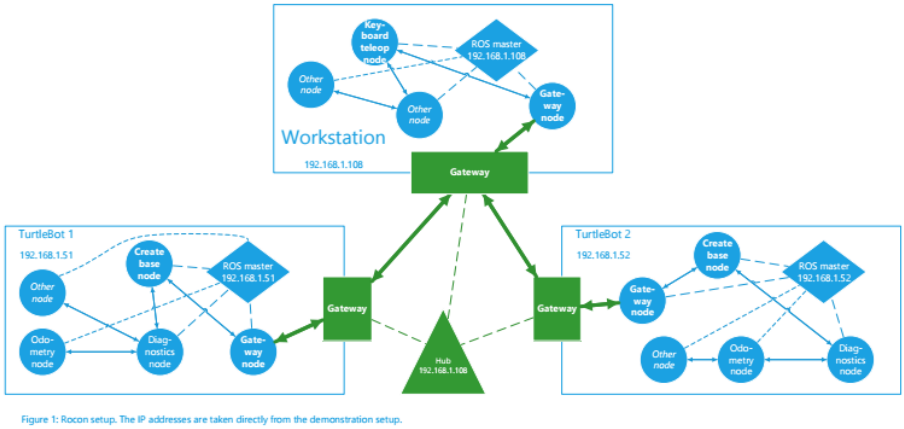

Figure 12: Rocon Framework, Hub is the triangle at the centre and Gateways or Turtlebot Masters other three rectangular blocks

ROCON framework has three basic building blocks- HUB, Gateways and Flipping nodes as shown. The HUB controls the flow of information i.e. keeps track of all the topics required by different masters and only sends the topic to a master which that master has subscribed for. The Gateways are interfaces between different masters and HUB. If a master wants to subscribe or publish a specific topic then it tells the HUB about it using Gateways. The Flipping nodes are the nodes that flip topics that a master has subscribed to. It is closely integrated with the Gateways since they serve similar purpose.

The people who have created ROCON package on ROS Indigo version, have actually introduced an abstract layered package which encapsulates the entire lower level working of ROCON. All one has to do is independently introduce your own package which contains robot functionalities into this abstract interface in the form of what they call as Rapps which stands for ROCON apps. Rapps are just packages which are taken as arguments in the ROCONs abstract package called as ROCON Capabilities package. After we create RAPPS, the Capabilities take the launch files of the Rapps and launch them in a file called concert-client. Launch.
The Rapps basically is a package of four files. They are as follows

1. Launch file
2. Interface file
3. Rapps file
4. Parameters file

The launch file is just like any other launch file where we launch all the nodes we desire and also we mention the re-mappings of the topics as they will be different in the HUB namespace (HUB - a master which regulates the communication of all the other masters and also is responsible to launch them).

An interface file is where you define the topics the master or the nodes within that master would subscribe or publish. This file is of YAML format as it is used during compilation period which is when you do Catkin make. We can also enter services if we are using client/server response method of two way communication.

A parameter file is the one in which we can import parameters to the Parameter server that is local to one particular master in a multi-master setup. Here we can initiate the variables with different values for example the robot name or node name or port number etc.

7.1.3 Planning and Navigation sub-system

In the navigation system we have is a pipeline of state machines. These state machines are event triggered. The pipeline proceeds in a systematic way as follows.

Figure 13: Event Triggered State Machine

The first state machine is where the central robot rotates around its own axis. This state machine continues until the person is found. Once the person is found using April tag mounted on his chest, the event flag is triggered and the state machine switches from R*otate* mode to *Flocking towards the person* mode.

Once the second state machine is turned on, the central robot moves towards the person detected while the other robots flock about it, thus all robots stay together as a flock. The *Flocking towards a person* mode ends when the central robot is about 1.5 meters away from

the person. When this happens, the event 2 flag is triggered and the state machine changes to *Align around the person* mode. In this mode the central robot goes to the point that is 1 meter away from the person at 0 degrees while the other two robots are given the coordinates at -30 and +30 degrees at 1 meter away from the person each.

When the robots reach the final coordinates respectively another event flag is triggered. This is when navigation ends and detection subsystem takes over, which again is event based.

Details of navigation subsystem

The project involves the use of an accurate navigation sub-system so that the swarms can easily move through the indoor environment and can place themselves at accurate positions to perform facial recognition collaboratively. There are three basic sub-blocks of navigation, namely Self-rotation and planning-navigation.

Self- Rotation

Turtlebot 1 will self-rotate about its own axis in clockwise sense with angular speed 12 degrees/second. This node will launch itself when there is no person of interest detected. This node subscribes itself to April Tag detector node which will publish April tag information as soon as it detects one. Once the node gets information from April tag subscription, it will call back a function where in it destroys itself and stops the rotation of the Turtlebot. The angular speed is set to a value such that April tag is detected even at a distance of 10-15 feet easily.

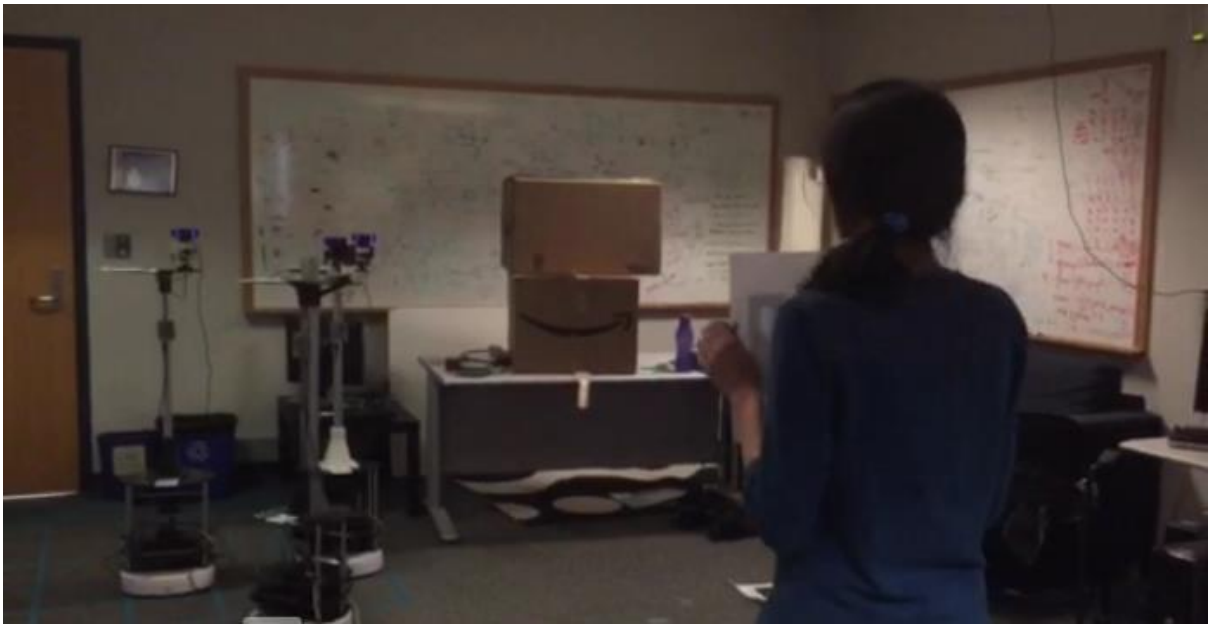

Figure 14: Central Turtlebot rotating and detecting human with April Tag

Human Detection

In the one robot- one human system, we do not require the entire map. Instead we just need the coordinates and orientation of the human relative to the Turtlebot 1. Fortunately the April Tag library provided by Professor Michael Kaess of CMU gives us a quite accurate relative position and orientation of the April tag. The information given by the April tag library is x, y, z and row, pitch, yaw. Additionally it also gives the Euclidean distance. The navigation node we set up uses this information from the April tag and tries to reduce the relative pitch and also reduces relative Euclidean distance to 100 cm. The result is that the Turtlebot 1(central Turtlebot) is 100 cm away from the human and is facing the human face to face.

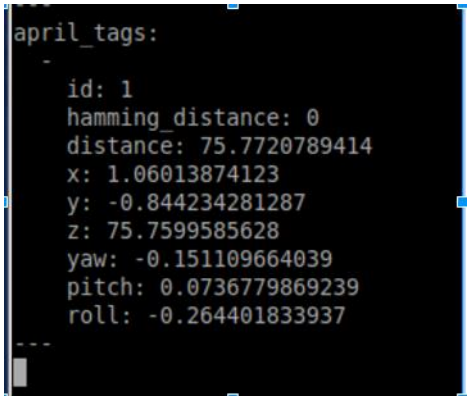

Figure 15: April tag topic output. It prints out x, y, z, yaw, pitch, roll, id and distance values

Flocking

In flocking, the Turtlebots move in a group such that they try to maintain a close proximity with each other and at the same time if they are too close, they tend to repel each other away. The flocking is a virtual potential field based algorithm mainly composed of three functions and they are as follows-
Cohesion**:** The velocity and the trajectory of the agent in question is only affected by those near it. Thus we define a certain R_attract as a circular approximation to the area within which an agent could affect other agent's velocity. In other words we only consider the agents within this circle with the agent in question as the center.
Repulsion: To avoid colliding with each other the agents should move away from each other if they happen to come too close to one another. So we define a R_Repel as the safeguard distance within which if the agent is found then it will experience a virtual repulsion force.
Alignment: In this function, if the agents are within R_attract and beyond R_Repel they tend to move in a similar fashion. Reynolds defined this function as each of the agent moving towards the center of mass of the entire flock.

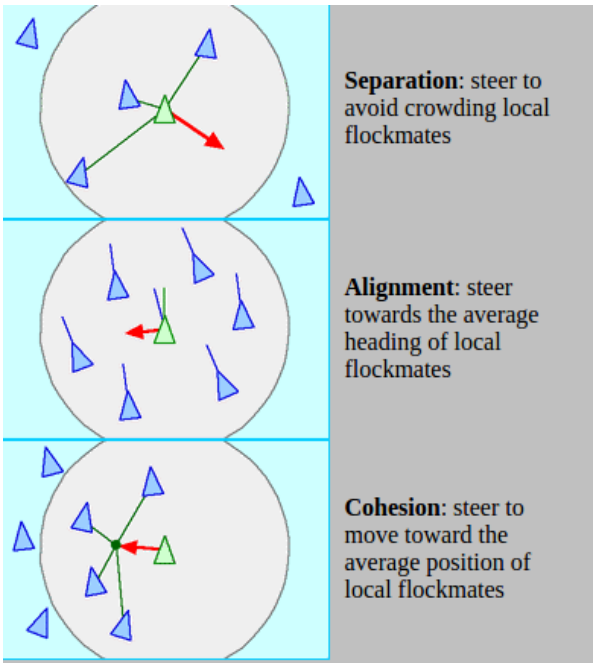

Figure 16: Flocking composed of 3 functions Separation (repulsion), Alignment, Cohesion

Figure 17: Turtlebots flocking towards the person.

Align Around the person

Once the central robot or Turtlebot 1 reaches a distance of 1.5m from the person, it triggers the event flag that switches the state machine from flocking to Align around the person mode. In this mode, the three robots are assigned the three coordinates around the person at -45, 0 and 45 degrees respectively at 1m from the person. Each Turtlebot then independently steer towards the nearest of the three points themselves. The final configuration that we achieved is shown below.

Figure 18: Turtlebots in process of aligning themselves around the person at -45, 0 and 45 degrees.

Overhead Localization

The localization of the system is done by global wide-angled cameras. There are 4 wide-angled cameras mounted on the ceiling. These cameras have a common region of overlap and an April Tag is stuck to the ground in each of these regions of overlap. The two cameras that view an April Tag in the region of overlap can know their relative transforms and in this way we can figure out the relative transforms between each of these cameras. If we know location of an object of interest relative to one camera, we can propagate using the relative camera transforms and find the object of interest's location in any other camera frame. Thus a global overhead localization system is set up. The object of interest in our case are various April Tags.

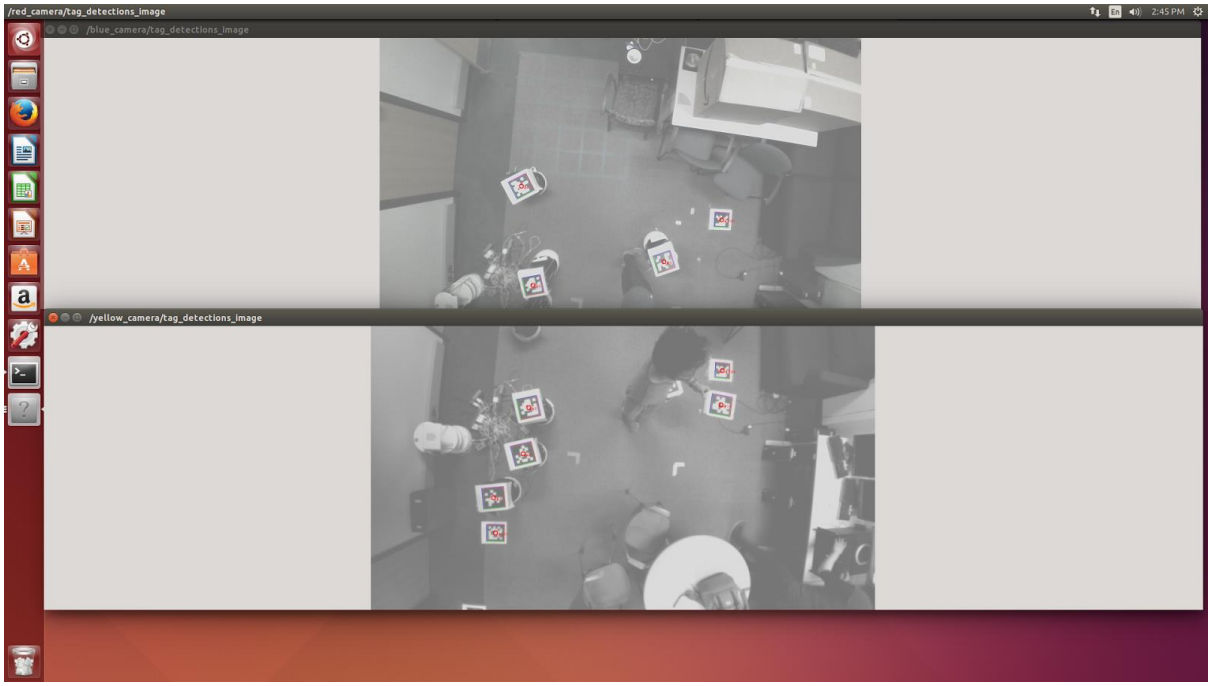

Figure 19: Overhead Localization of two nearby cameras detecting the April Tags on Turtlebots

7.1.4 Detection subsystem

Facial expression detection

In the detection subsystem the pipeline is event triggered which is same as that of navigation system. This sub-system is highly dependent on the software IntraFace. The IntraFace software gives us the following as topics

1. Probabilities of 6 Emotions – Happy, Neutral, Sad, Anger, Disgust, Surprise.
2. Head Pose in quaternion form
3. Pixel coordinates of the 49 facial action units or simply facial features.
4. Eye gaze in quaternion form

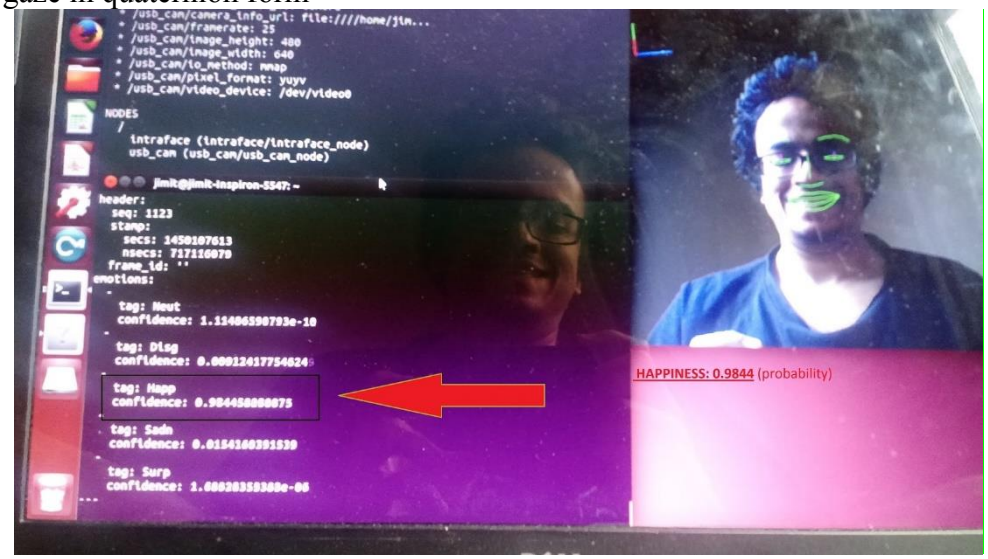

Figure 20: The person of interest is smiling and the "happiness" reading is 0.98

As soon as the detection subsystem's state of Align around a person sends the flag that the Turtlebots are in position at -30,0 and 30 degrees from the person of interest at 1 meter distance, the flag activates the detection sub-system. This sub-system comprises of different nodes that run on each of the Turtlebots which publish both emotions and head pose. All this message of

emotion and head pose start getting published after they receive the flag from navigation sub-system. These published values are then received by emotion comparator which takes in both the head pose and the happy emotion value. The happy emotion as shown in the figure is the 3rd element of the Emotion array. This node takes in both head pose and happy emotion value from all the Turtlebots and calculate a cost function based on both the values. The more closely aligned is the head of the person and the more the happy emotion value is of the person, the more is the cost function value. This node compares the cost function values of all the 3 Turtlebots and sends a flag to the Turtlebot which has the highest cost function, which means this Turtlebot has detected the best smiling probability of the person and also has better view of that person's face.

Once the flag is sent to the corresponding Turtlebot, that Turtlebot then clicks the photo of that person.

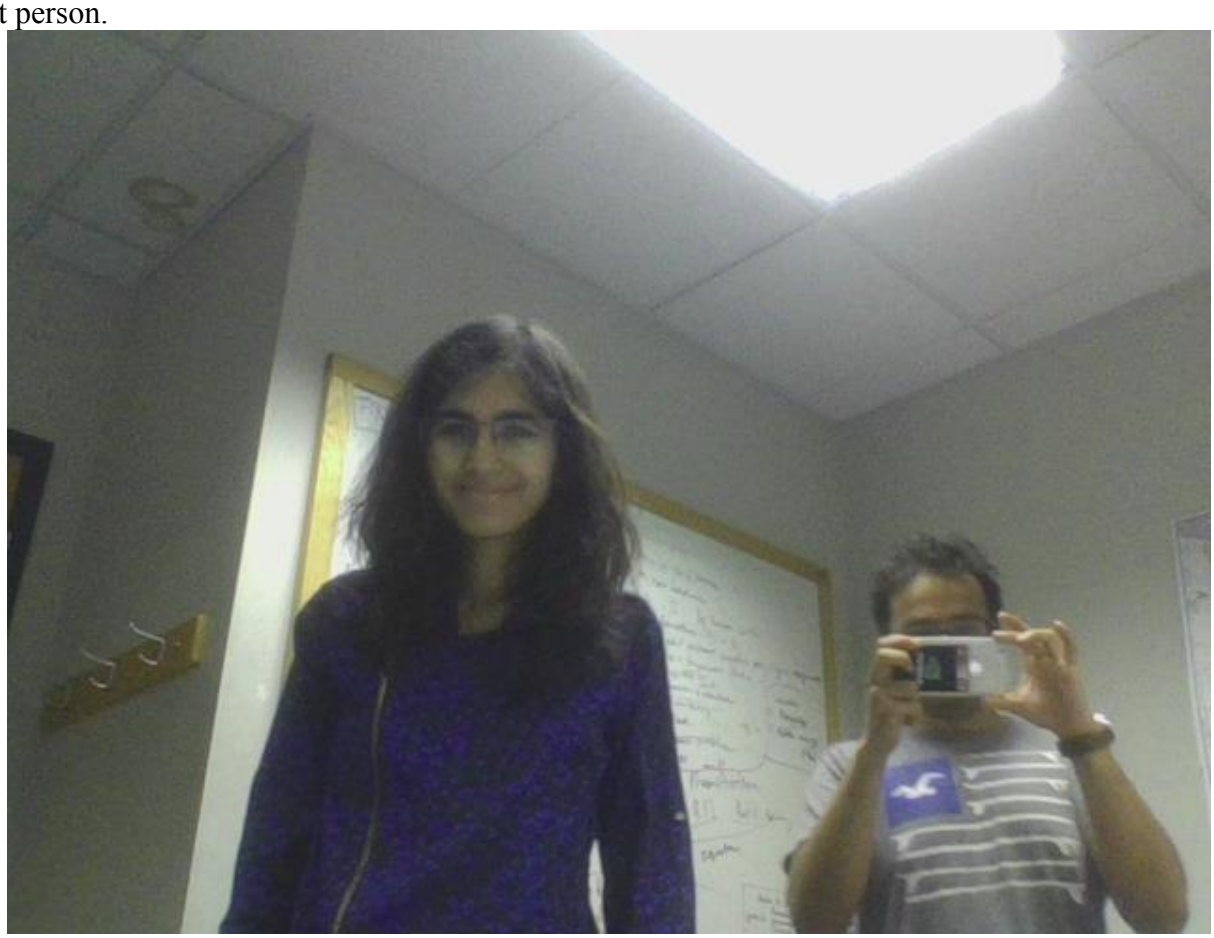

Figure 21: Final Photo taken by the Turtlebot with best cost function value

Face tracking control with pan-tilt unit

We track the person of interest by using the coordinate of his/her nose to represent the whole face. The algorithm of face tracking is based on a proportional controller, which covert the width and the length of the image into corresponding angle of pan and tilt servo motors. There's a serious distortion of data range. We constrain the data and convert it to ordinary range using the algorithm similar to Celsius/Fahrenheit conversion. To top the function, we cut the image into three section on the horizontal and vertical direction and form a rectangle of 100*100 pixel, if the nose exceed the region the proportional controller will kick in. Otherwise the motors will remain the same position, thus dramatically reduce the vibration problem of the servo motors.

7.2 Modelling Analysis and testing

The analysis and testing was done keeping the requirements in mind. We rewrite the requirements again. Since the major two subsystems that we were to be verified were

- Human Detection and Navigation
- Face and Expression Detection

7.2.1 Subsystem 1: Human Detection & Navigation

Requirements to be fulfilled

- Detect human in the vicinity
- Approach the human once detected
- Move as a flock with a speed of 15 cm/sec
- Stop at 1 meter away from the human
- Align around the human at -45, 0, +45 degrees angle

The following questions were framed as which if answered would lead to the fulfilment of the above mentioned requirements. If all of these questions are answered "yes" (in performance) by the subsystem then we could confidently say that the subsystem works.

Testing Criteria for Human detection and navigation subsystem:

- Does the central robot rotate in place?
- Does the central robot stop rotating in place instantly if April tag is detected for at least 70 percent of the times?
- Does the robot flock move toward the April tag?
- Does the central robot stop at 1 meter away from the Human?
- How much does the final distance deviate from 1m?
- Do the robots align themselves around the human?

7.2.2 Subsystem 2: Face & Smile Expression Detection

Requirements checked

- Detect Face
- Pan tilt units tracks the face using head pose estimate from IntraFace
- Accurate smile expression detection and head pose detection
- Best photo is clicked

Similarly for this subsystem following questions were framed as which if answered would lead to the fulfilment of the above mentioned requirements. If all of these questions are answered "yes" (in performance) by the subsystem then we could confidently say that the sub-system works.

Testing criteria for face detection and expression detection

- Does IntraFace detect face at least 80 percent of the time
- Do the pan tilt units adjust themselves such that face is in center of the frame?
- Do we get expression output every time?
- Is the best photo on the basis of expression and head pose clicked every time?
- Time taken to do the above task?

7.2.3 Complete System

Requirements checked

- All the requirements mentioned for the above sub-systems
- Face Detection starts only after the flock aligns itself around the person
- The same person is not clicked again
- The system restarts after one successful photo

Testing criteria for the complete system

- Does face detection start after the robots have reached the person of interest?

- Do the pan-tilts start tracking only when the robots have stopped?
- Does the swarm not click the photo of the same person again?
- Does the system restart autonomously after each successful run?

7.3 Spring Validation Experiment evaluation

Based on the questions framed using the requirements of each subsystem in the last section, we prepared a table of criteria which would affect the answers to those questions depending upon the success and failures to meet them.

The table is designed such that if the criteria is met, then we mark it as 1 or we mark it as 0 otherwise. If the criteria is met partially and is within the acceptable value then we mark it as 0.5. The total score of each iteration is taken and divided by the number of criteria. This would give us the success and failure probability. If this probability is more than what we promised in performance requirements then the system is delivered as promised but if not then there is certain improvement that has to be done.

For the complete system, we performed 10 iterations with different initial conditions and restarting the system entirely. By different initial conditions, we placed the Turtlebots at random positions in the room and human was placed randomly a with the April tag in the room.

Run No	Task	Correct April Tag detection	Correct Flock formation	Correct flock navigation	Correct angular arrangement	Correct face tracking	Correct photo capture	Correct relaunch of system
	Criterion	Detects the first tag and stops	1. Detects human 2. forms flock	1m stopping distance	(-45,0,45)	1.Cameras at correct initial positions(facing the person, motors at 90 deg) 2.Both the servos move with the person	1. 'Smile please' after person detection 2. Only smiling photo 3. Logically correct camera clicks the photo 4. 'Nice smile' after photo clicked	1. Same person not detected again 2. System restarts after one successful photo
	Grade scale	Yes:1 No:0	Yes:1 No:0 Sequence not followed:0	Stops at : 1m: 1 0.9-1 m : 0.5 Does not stop:0	Yes:1 No:0 Approx yes:0.5	Yes:1 No:0 Sequence not followed:0	Yes:1 No:0 Sequence not followed:0	Yes:1 No:0
1		1	1	1	0	1	0	0
2		0	1	1	0	1	1	0
3		0	1	1	1	0	1	1
4		1	1	1	1	1	0	1
5		1	0	1	0.5	0	1	1
6		1	1	1	1	1	1	1
7		1	1	0.5	1	1	1	1
8		0	1	1	0.5	1	1	1
9		1	1	1	1	1	1	1
10		1	1	1	1	1	0	1
Desired Success rate		70	80	80	70	80	80	70
Achieved success rate		70	90	95	70	80	70	80

Table 5 SVE Performance Matrix; As can be seen from the performance matrix, most of the performance requirements were fulfilled in the complete integrated system

7.4 Strong and weak points of current system

7.4.1 Strengths

Accurate reading of global camera

We worked on interfacing the four global cameras with the main PC and then performing intrinsic calibration for each one of them to get the camera parameters. We tried to get as many sample points for the calibration as we could so that the cameras give accurate localization readings.

Good functionality of flocking and navigation behavior

The flocking algorithm and the Rapps for the ROCON are well combined for the three Turtlebots. These Rapps allowed the setup to subscribe to April Tag locations and odometry readings for each of the Turtlebots and publish on their base velocities. To try out this complete framework for navigation, we used the other three Turtlebots in the Advanced Agents Lab with netbooks.

Full combination between single person detection frame and pan-tilt face tracking function and voice command

We used the x coordinate of April Tag to replace the x coordinate of the nose position. The x coordinate of the April Tag is coming from the original frame, and it's not processed by Intraface. Only the narrow frame with the target person needs to go through the Intraface. After finding the target person, a series of voice commands are sent out until the person smiles.

Integration of subsystems of IntraFace, ROCON, Navigation, Flocking and the arrangement around the target person

We have implemented the integration of the subsystems of IntraFace, ROCON, Navigation, Flocking and the arrangement around the target person. The operating system of perception subsystem is Ubuntu 14.04. And we installed 64 bit version of IntraFace which can be controlled by ROS commands on Ubuntu 14.04. In this way, we can start facial expression recognition with IntraFace and simply extract the position of interest points, the percentage of each facial expressions as well as other features with ROS commands.

The process is fast without much delay

One of the biggest strength is that the perception subsystem is very fast, and there seems to be no delay for the whole process, which means that the pan-tilt unit can track the person's face and the IntraFace can detect the person's facial expression in a very short time.

4.2 Weaknesses

Chromebooks crash

One of the biggest weaknesses of the system is the reliability on Chromebooks. The Chromebooks sometimes crash if their batteries are allowed to drain out completely. In such a situation, we have to re-install everything from scratch. This takes a lot of time.

Lighting conditions

The system requires optimal lighting conditions to work. In the absence of proper lighting, both the navigation and the detection sub-systems are affected.

8. Project management

8.1 Project schedule

We changed our approach on tracking the project schedule. Earlier we tried a few softwares such as Microsoft project, Slack projectmanager.com and some more. However, nothing amongst them turned out to be really user friendly. Also, we observed that the Gantt charts were not of much help during our team meetings. They occupied the half of the screen and resulted in unnecessary confusions being created. Hence, we decided to omit the Gantt charts from the tracking sheets. Another change was tracking only the high level tasks and not the minute once. Due to the heavy course load in the MRSD curriculum, it became really difficult to track all the minute details. With these changes, modified our schedule tracking mechanism to using a simple google sheet, shared within the team which is shown as follows:

ID	Name	Responsibility	Duration	Start	Finish	Time of Completion	% Completion
2	**Mechanical Work**						
2.1	Elevation Mechanism CAD design	Rohit	2 days	07-Nov-15	08-Nov-15	11.59pm	100
2.2	Elevation Mechanism fabrication	Rohit	3 days	08-Nov-15	10-Nov-15	11.59pm	100
2.3	PTZ + Elevation assembly on TB2	Rohit	1 day	11-Nov-15	11-Nov-15	11.59pm	100
3	**Human detection**						
3.1	Turtlebot 1 self rotation-Mannequin 1 AprilTag detction algorithm in ROS	Gauri, Jimit	3 days	09-Nov-15	15-Nov-15	11.59pm	100
4	**Planning and Navigation**						
4.1	Mannequin 1 position estimation algorithm	Gauri, Jimit	2 days	17-Nov-15	25-Nov-15	11.59pm	100
4.2	Mannequin 1 odometry using Webcam	Gauri, Jimit	7 days	25-Nov-15	28-Nov-15	11.59pm	100
4.3	Apriltag detection algorithm	Gauri, Jimit		28-Nov-15	30-Nov-15		100
4.4	Turtlebot 1 successful navigation to Mannequin 1 (Accurate 5 Ft distance)	Gauri, Jimit	7 days	30-Nov-15	01-Dec-15	11.59pm	100
5	**Smile face detection**						
5.1	Face detection using webcam & IntraFace/MATLAB for Mannequin 2	Tiffany, Sida	4 days	07-Nov-15	10-Nov-15	11.59pm	100
5.2	Smile Expression detection using IntraFace and Webcam for Mannequin 2	Tiffany, Sida		11-Nov-15	18-Nov-15	11.59pm	100
5.3	PTZ camera face scan algorithm in Arduino and Intraface	Tiffany, Sida		19-Nov-15	25-Nov-15	11.59pm	100
6	**Image fusion**						
6.1	Photo capture on smile detection algorithm in ROS	Tiffany, Sida		25-Nov-15	01-Nov-15	11.59pm	0
6.2	Algorithm to obtain image from TB2 on workstation	Jimit		03-Nov-15	10-Nov-15	11.59pm	0
7	**Integration**						
7.1	Integrate Human detection & Planning-navigation	Gauri, Jimit		23-Nov-15	01-Dec-15	11.59pm	100
7.2	PTZ unit mounting on TB1 & testing	Rohit		12-Nov-15	20-Nov-15	11.59pm	100
7.3	PTZ camera + Face detection algorithm	Sida, Tiffany		26-Nov-15	01-Dec-15	11.59pm	100

Table 6 Project Schedule for Fall 2015

ID	Name	Responsibility	Start	Finish	% Completion	
1	**Parts ordering**					
1.1	Trade study (Chromebooks, cameras)	Rohit	20-Jan-16	22-Jan-16	100	PR7
1.2	Order chromebook and camera	Rohit	23-Jan-16	23-Jan-16	100	PR7
1.3	Testing with the system and order 2 more sets	Rohit	31-Jan-16	08-Feb-16	100	
2	**Mechanical Work**					
2.1	Problem analysis in current assembly (rev 1.0)	Rohit	07-Feb-16	10-Feb-16	100	PR9
2.2	Elevation Mechanism CAD re-design (rev 2.0)	Rohit	13-Feb-16	22-Feb-16	100	PR9
2.2	Elevation Mechanism fabrication	Rohit	24-Feb-16	02-Mar-16	100	PR 10
2.3	PTZ + Elevation assembly on Turtlebots	Rohit	12-Mar-16	14-Mar-16	100	PR 10
3	**Detection**					
3.1	Testing human detection + face detection using same camera	Gauri, Jimit	23-Jan-16	25-Jan-16	100	PR7
6	**Image fusion**					
6.1	Algorithm to obtain image from TB2 on workstation	Jimit	25-Jan-16	26-Jan-16	100	PR7
6.2	Photo capture using multiple turtlebots	Jimit,Sida	08-Mar-16	15-Mar-16	100	PR 10
4	**Planning and Navigation**					
4.1	ROCON simulation	Jimit, Tiffany	24-Feb-16	08-Mar-16	100	PR 10
4.2	Physical ROCON setup	Gauri, Jimit	08-Mar-16	14-Mar-16	100	PR 10
4.3	Global camera setup	Gauri, Jimit,Sida	08-Mar-16	10-Mar-16	100	shown in PR9
4.4	Global camera calibration	Gauri, Jimit,Sida	10-Mar-16	14-Mar-16	100	shown in PR9
4.5	Apriltag detection algorithm	Gauri, Jimit,Sida	16-Mar-16	23-Mar-16	100	PR 11
4.6	Turtlebot localization	Gauri, Jimit,Sida	16-Mar-16	23-Mar-16	100	PR 11
4.7	Flock formation	Gauri, Jimit,Sida	24-Mar-16	31-Mar-16	100	PR 11
7	**Integration & testing**					
7.1	Integrate mechanical, detection & Planning-navigation	ALL	03-Apr-16	10-Apr-16	100	SVE
7.2	Testing the whole system	ALL	13-Apr-16	27-Apr-16	100	SVE

Table 7 Project Schedule for Swarm and Planning Subsystems for Spring 2016

8.2 Budget

8.2.1 Purchased Parts List:

S.No	Sub-system	Item(Qty.)	Obtained from	MRSD Budget used
1	All	Acer Chromebook C720(3)	Amazon	358x3
2	Navigation	Turtlebots(3)	Advanced Agents Lab	0
3	Mechanical	Aluminium Rods(3)	FRC	0
4	Mechanical	Pan-tilt mounts(3)	3D-printer (MRSD Lab)	0
5	Mechanical	Servo motors(6)	MRSD Lab Inventory	0
6	Detection	Logitech C920 Camera(3)	Amazon	67x3
7	Detection	Intraface software	Human Sensing Lab	0
8	Detection	April Tags	Advanced Agents Lab	0
9	Detection	Arduino(3)	MRSD Lab Inventory	0
10	Mechanical	Fitments, Motors & Casings	Multiple Sellers	530
11	All	Shipping & misc.	Multiple Sellers	245
			TOTAL SPENT	**2040$**
			% EXP/BDGET	**51.00%**

Table 8 Budget Analysis

8.2.2 Budget statistics:

Total Allocated Budget = 4000$

Total Budget used /Total budget spent till date = 2040$

Total percentage used for the project = 51%

8.2.3 Budgeting process, successes and failures:

After doing the trade analysis for the required components required for the project, we decided to prepare a budget for the parts to be ordered as above. We had also differentiated them as immediate orders and future orders. As decided in team meeting, we ordered 1 quantity of the expensive items such as cameras and the Chromebook, with the plan of doing successful testing with the system before ordering multiple quantities as part of the risk management. After receiving a single quantity of these items, we checked their compatibility with the other components in our system. This strategy worked out very well as some ill components were identified beforehand. As a result, we did not have any bad or incompatible parts in our orders and budgeting. However, one of the Chromebook laptops crashed just before the SVE and we had to spend a lot of time in resetting it, due to which the system integration got delayed.

8.3 Risk Management

Risk management is very important since this tool helps us to foresee the problems that may arise and how they will affect the project performance and functionality. As it is rightly said "Hope for the best but prepare for the worst", risk management helps us prepare for the worst. The team identified the potential problems that could occur and marked them as malignant or benign based upon the consequences and likelihood of that problem if it occurred. The following were the potential risks identified and also the corresponding strategies or steps proposed in order to reduce and mitigate the effect of that risk. The risks have also been pointed out in figure 22 in the form of risk management chart.

1. **Risk #1: Noisy and blurry video stream due to robot moving.**

Risk Type: Severe: Likelihood was more since robot would be always moving and capturing video. Since expression detection involves clear video stream to capture the micro-facial interest points it was pertinent that vibration caused to the camera due to robot movement be minimized so as to get high efficiency of expression detection software.

Risk mitigation strategy: The source of vibration was due to the less strong base of the elevation axis. The design of the base was modified to make it stronger and firmer. In this way the vibrations were reduced significantly. The risk is now benign as the system can now perform without much difficulty and thus meeting the performance requirements.

2. **Risk #2: Battery drain in Turtlebots.**

Risk Type: Moderately severe: The battery life of Turtlebots is less (30-40 minutes from full charge), repeated testing cannot be done. This is a wastage of the most precious resource of our team which is time. We have faced events that lead to frustrations.

Risk mitigation strategy: The reason that Turtlebots have low battery life is because the Turtlebots we were working on are 5 years old. This has reduced the battery life considerably with repeated timely charging and discharging. The lab sponsor thus decided and ordered a new set of batteries as well as spares such that in the event of low battery, one can replace a battery and put that discharged battery on charging. Our efficiency doubled since the arrival of the new batteries.

3. **Risk #3: Single Robot failure**

Risk Type: Moderately severe: If a single robot fails then the performance of the system goes down. Although the likelihood of this risk is relatively low, the consequences could be high. Even if other robots work perfectly fine, failure of a single robot could lead to undesirable effects in terms of performance.

Risk mitigation Strategy: The swarm subsystem should be planned and executed such that in the event of a single Turtlebot breakdown, the performance of the system as a whole should not be affected to a great extent. Thus, we implemented ROCON wherein each Turtlebot was a master of its own. The only centralized functionality in the system was hub that controlled the exchange of information on the network.

4. **Risk #4: Extra Payload**

Risk Type: Moderate: The Turtlebots have a limited payload capacity of 7-9 lbs and beyond that they will not achieve the speed and movement we desire. Thus it is imperative that the weight of the entire pan- tilt plus the camera with elevation axis should not exceed the above mentioned limit.

Risk mitigation strategy: There are many light weight HD camera available in the market which are within our budget as well. We have included weight of camera as an important parameter during camera trade study analysis. The pan tilt design also is designed such that we use minimum material but at the same time provide support and stability to the mounted camera. The material to be used will be aluminium.

5. Risk #5: Crashing of Chromebooks

Risk Type: High: We have completely removed the ChromeOS from the Chromebooks. All the three Chromebooks are now Linux-based. Due to this, the machines have crashed multiple times and we have to install everything from the scratch.

Risk mitigation strategy: To mitigate this risk, we prepared our personal laptops as the backup machines for the Turtlebots. We also take care that the batteries of the Chromebooks do not drain out completely to avoid crashing.

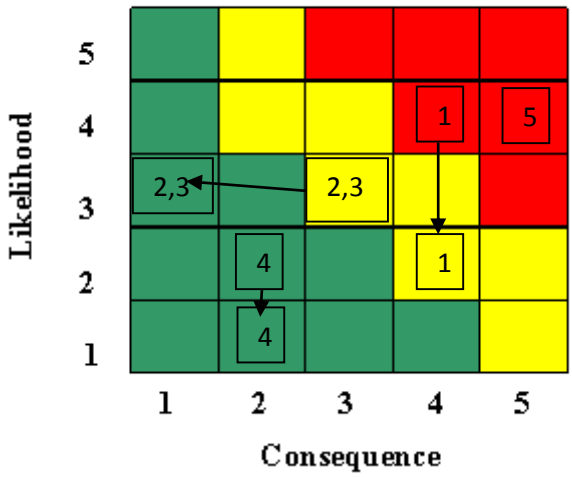

Figure *22* **Risk management table: The above figure depicts the numbers of the risks mentioned in the description on previous page. The arrow suggests that the risks were initially where the arrow starts and where the risks now stand after mitigating them, represented by the arrow heads.**

9. Conclusions

9.1 Lessons Learned

1. **For those very important things, always prepare a plan B**
   Prepare a plan C for plan B, because everything may crash down. If nothing crashes, something will automatically crash to fill the void.

2. **Communication**
   Always communicate with your teammates. Don't expect others will know what you are thinking. And don't start doing your job without letting other people know what you are doing. Sometimes not doing something is as important as doing something.

3. **Try to see the future. Find out the risks**
   Then you can prepare early to face the risks.

4. **Do not expect magic from day 1**
   Magic wouldn't happen and the future will not be changed if we don't change our behavior. We learnt to be practical and start things as early as possible. After starting, we realized all kind of odds could stack up against us. Some of the problems didn't reveal themselves unless we started, like the unexpected crash down of Intraface and noise problems we faced in moving data.

5. **Keep everyone on the same page**
   The miscommunication happened at the very first time of the team meeting. Five people had five different ideas. And since we tend to not speak out and disagree with teammates, it took a while until we finally reach the agreement. But it was worth it, after we were on the same page, we could reach the goal at the full speed without hesitation.

6. **Clarity is very important**
   The clarity is the key point of whether the communication is successful or not.

7. **Do not reinvent the wheel**
   This lesson was learnt when we were trying to build our own human detection and face tracking function, and then we realized that Intraface already did that for us. If we would have found out that earlier, we could have saved some time.

8. **Take small steps to achieve required goals**
   Some of the work seems impossible to be done. Cutting down tasks to small sub tasks always helps.

9.2 Future Work

1. **Eliminate Overhead Localization-** The project's navigation subsystem was solely dependent on overhead cameras. This requires initial setup for the event which is not very optimal. One could do this by using camera to perform visual odometry and fuse that with wheel odometry and IMU readings to get approximate location.

2. **Remove April Tag from Human Detection –** There are many ways to detect humans some of which are machine learning based algorithms such as Histogram of Gradients method.

3. **Make robots appealing aesthetically –** Since the robots are intended to be used in social occasions with people around, they should appear more friendly and approachable in nature.

4. **Use camera features such as auto-focus and Zoom-** One thing we noticed that our output was based on smiles but sometimes picture of the person would be blurred and robot has to come closer to the person to capture his expressions. One could simply zoom instead of going closer to the person and also employ auto- focus for better quality.

10. References

1. http://www.humansensing.cs.cmu.edu/intraface/
2. http://wiki.ros.org/rocon
3. http://shop.soloshot.com/
4. https://en.wikipedia.org/wiki/Lidar
5. http://www.roborealm.com/forum/index.php?thread_id=2681
6. http://www.amazon.com/Panasonic-HC-V110-Weight-Digital-Camcorder/dp/B00AW54YWS
7. http://wiki.ros.org/rocon_app_utilities/Tutorials/indigo/Introspect%20Rapps
8. http://kernel.ubuntu.com
9. https://en.wikipedia.org/wiki/Flocking_(behavior)
10. http://gamedevelopment.tutsplus.com/tutorials/the-three-simple-rules-of-flocking-behaviors-alignment-cohesion-and-separation--gamedev-3444