Augmented Reality Debugging System for Swarm Robotics

Initial Report

Alistair Jewers

February 2017

Abstract

Due to their complex and dynamic nature, debugging swarm robotic systems is a challenging task. This report describes a project which will aim to develop a tool for monitoring a robot swarm in real time and providing a real time video feed of the swarm's environment, overlaid with graphical representations of relevant data in an '*augmented reality*' fashion. The tool should enable the user to debug the system more effectively. An overview of the project context is given, and a summary of relevant elements of the literature is provided. The project aims are formalised and the objectives stated. A specification for a software application which will satisfy the aims is given, and the development tasks required to implement this application are identified. Finally a plan for completing the project within the available time frame is presented, and the practicalities and timing risks associated are considered.

Contents

1 Project Overview and Aims

Swarm Robotics is the name given to the nascent field of study focusing on the use of concepts derived from the study of social insects, such as ants or bees, to design and implement behavioural algorithms for multi-robot systems [1]. These behaviours should allow a group of relatively simple robots to achieve a more complex, emergent behaviour through cooperation. The broader area of study, without the robotics focus, is referred to as Swarm Intelligence (SI), and is described by Dorigo & Birattari as the *"discipline that deals with natural and artificial systems composed of many individuals that coordinate using decentralized control and self-organization"* [2], with examples including insect colonies, fish schools, and flocks of birds. Whilst the details of this complex area of study are outside the scope of this report, it is of importance to the nature of the project to note that one of the key aims of swarm robotics is decentralised control. To this end, in a swarm robotic system you would not expect to find any master controller or central decision making unit. Instead, each robot acts based purely on information available locally, and no point in the system is aware of the current state of all the robots. Also of note is the fact that the state of a robot may change rapidly over time, and be dependent on a large number of environmental or outside factors. Considering these two statements together, it becomes readily apparent that debugging a swarm robotic system may present an enormous challenge, as the state of the system will be complex and rapidly changing, and no single point in the system can provide the user with the complete state information.

This project, entitled '*Augmented Reality Debugging System for Swarm Robotics*', focuses on the design and implementation of a computer application for monitoring and debugging swarm robotics systems in real time. This will include the use of an existing video based tracking system to monitor the robots' positions. The robots will communicate information regarding their current internal state, sensor readings, and other decision critical data to the computer wirelessly. Graphical representations of the spatial elements of the robots' data will then be overlayed on top of the video feed, whilst non-spatial data will be represented in other forms. By fusing the data from these two sources and presenting it to the user in a combination of graphical and textual formats, the software will aim to allow the user (most likely the researcher running the swarm robotics experiment) to isolate faults in the system more quickly, and determine if the nature of a problem is related to the behaviour under test, or another factor such as sensor/actuator malfunction, incorrect state transition, *etc.* Another aim of the project is to provide this debugging facility in a highly modularised way, which can be incorporated into a swarm robotics system with relative ease. The system will initially target the widely used e-puck [3] robotics platform, but will aim to be designed in a way that allows support for other robots to be incorporated without modifying the core system. Figure 1 shows a logical representation of the expected system architecture, utilising the e-puck platform.

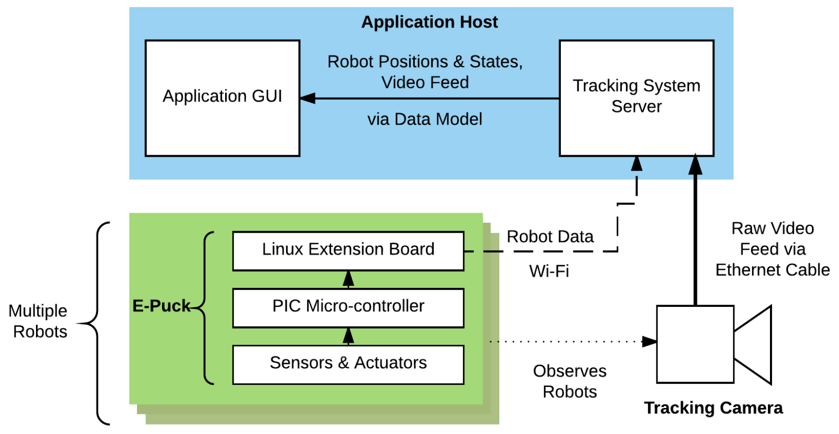


Figure 1: Expected general system architecture

2 Literature Survey

A number of key areas of literature have been identified as relevant to this project. An understanding of the fundamental concepts of Swarm Robotics, and to a lesser extent Swarm Intelligence, will be key to producing an application that is useful in practice. An overview of the core concepts as well as some key publications are presented in Section 2.1. A deep understanding of the technical details of specific swarm systems, such as specific behavioural algorithms or implementation details, is not a priority for this project, as the application aims to be more broadly applicable to a wide range of swarm systems. Hence, emphasis is therefore placed on the general classification of swarm robotic systems, relevant problem domains, and recurring concepts, so that the software might better serve researchers in the field.

A relevant area of current research is Human-Swarm Interaction (HSI) - determining how best to interface humans with swarm systems in various roles. This presents two key challenges: the first being how to allow a human operator to provide input and direction to a swarm system without breaking the decentralised control paradigm [8], and the second being how to retrieve data from a swarm system in a coherent form that a human might easily understand [9]. This project will be focusing on the latter problem in the context of debugging a swarm system, with the human in a developer role, rather than a user or researcher role where they would be focused on the systems intended output. Despite this perspective difference, HSI research remains highly relevant and must inform the design of the application. An overview of the Human-Swarm Interaction literature is presented in Section 2.2.

Recent advances in virtual-reality (VR) and augmented-reality (AR) technologies have led to an increased interest in using these technologies in conjunction with robotics. AR especially presents a powerful tool for HSI, as an augmented space can be readily understood by both humans and robots. Research relating to the use of AR with robotic systems is summarized in Section 2.3, alongside a summary of the work currently existing which utilizes these concepts in the context of multi-robot systems, and other real time, graphical debugging systems which bear similarities to the aims of this project.

2.1 Swarm Intelligence and Swarm Robotics Overview

Sahin [1] presents a summary of the key concepts of swarm robotics, and attempts to offer a coherent description of the topic. He notes that a key difference from other multi-robot systems is the lack of centralised control, and the idea that desired behaviour should emerge from simple local interactions between robots, and between the robots and their environment. He also notes some of the key motivators behind Swarm Robotics research, stating that a swarm robotics system would ideally have "*robustness*", "*flexibility*" and "*scalability*" [1]. Robustness refers to the swarm's ability to continue to function should one or more individual swarm members suffer a failure of some kind. Flexibility refers to the swarm's ability to adapt to changes in the environment without the need for re-programming. Scalability describes the idea that a swarm should be functional at a range of sizes, and that ideally the number of robots in the swarm could be increased or decreased depending on the demands of the task. Sahin [1] goes on to describe several classes of application where Swarm Robotics systems might be well suited. Tasks that cover a region could benefit from a swarm's ability to distribute physically in a space according to need. Dangerous tasks could benefit from the relative dispensability of individual robots in the swarm; should one be damaged or destroyed the swarm could continue to function, and it would be less costly that the loss of a single, complex, expensive robot. Tasks requiring scalability are good candidates, as discussed before, and tasks that require redundancy are also highlighted, as swarm systems should have the ability to degrade gracefully, rather than suffering a single catastrophic failure. Through this generalisation of the application areass, insight can be gained into the kinds of work swarm robotics researchers are likely to be doing, and this should inform the design of the application. This paper [1] provides a coherent, succinct overview of the field, and although it is now over a decade old the concepts covered remain relevant. The paper also contains a wealth of further reading, including papers on developing specifc behavioural paradigms such as self-organisation [4] and path-formation [5], which give insight into the kinds of information and data that swarm robots use to make decisions, and that a swarm researcher might therefore be interested in monitoring. Beni [6] presents a relatively informal but useful overview of the terminology used in the field, which may serve as useful additional reading to Sahin's overview.

The book '*Swarm Intelligence: From Natural to Artificial Systems*' written by Bonabeau, Dorigo and Theraulaz [7] provides in its introductory chapter a good overview of the biological concepts and animal behaviours which inspire the field of swarm intelligence. The later chapters provide a detailed look at several of these behaviours, and how mathematical models and algorithms can be derived from them. Although more detailed than this project requires, an understanding of these behaviours and models can offer insight into what information the application might need to expose to the user to allow them to validate the correct operation of a system based on these concepts.

2.2 Human-Swarm Interaction

In their paper '*Human Interaction with Robot Swarms: A Survey*' [8] Kolling at el. begin by noting the lack of research into methods for interfacing humans and robot swarms. They suggest that real-world applications for swarm robotics systems are now within reach, and that discovering effective methods for allowing humans to control and/or supervise swarms is a key barrier to realising these systems. The paper [8] provides a detailed analysis of human swarm interaction from a number of different perspectives. Of relevance to this project is the statement on page 15 that "*Proper supervision of a semiautonomous swarm requires the human operator to be able to observe the state and motion of the swarm, as well as predict its future state to within some reasonable accuracy*" [8]. Considering that swarm supervision and swarm debugging are highly comparable tasks - both involve observing the swarm whilst performing its task and determining the validity of the behaviour observed - this statement lends credence to the aims of this project. The proposed application would allow the state of the swarm, including the internal state of individual robots, to be observed simultaneously with the physical positions and motions of the robots within their environment. The paper [8] goes on to suggest that by observing the swarm over time the human operator will be able to provide '*appropriate control inputs*'. In the case of this application, rather than providing control input, the human operator will be seeking to identify faults, and provide appropriate corrections to the system, however the concept of state visualisation remains relevant.

Rule and Forlizzi [9] present a thorough examination of the complexities of human robot interaction (HRI) when dealing with multi-robot (and multi-user) systems. Much of the paper focusses on control methods, which are not directly applicable to this project, however Section 2.4 titled *Salience of Information* discusses the task of designing interfaces for displaying information about multi-robot systems to a human operator in a manner which is both information dense and rapidly understandable. The authors note that the use of colour has been shown to improve interface readability [10], and that the brain has been shown to process text faster than images [11], hence complex icons should be avoided. These ideas should be incorporated into the design of the application user interface for this project. A range of different designs could be explored, including finding a balance between the amount of information displayed graphically, and the amount displayed textually, and deciding whether to use colour to differentiate between individual robots, or to differentiate between different types of data, or a combination of both.

2.3 Augmented Reality and Other Graphical Systems for Robotics Observation and Interaction

Collet and MacDonald [12] describe in detail the difficulties in debugging robotics systems, and how Augmented Reality (AR) in the context of robotics and HRI presents a uniquely useful tool for overcoming some of these difficulties. The authors identify that the difficulties in developing and debugging robotics applications when compared to traditional software arise from either the environment of the robot - which will often be "*uncontrolled*" and "*dynamic*" - or from the mobile nature of the robot. Because the environment a given robot operates in is a real world space, the level of control that can be exerted over it by the researcher or operator is inherently limited [12]. The environment may therefore change over time, exhibit imperfections, and include other time-varying elements. A robot is a physical actor and will likely experience dynamic change in its sensor readings and its relationship to the environment over time. This is especially true for mobile robots, whose position and orientation will change over time. The behaviour of the robot often largely depends on these highly variable factors, and therefore replicating a given behaviour exactly becomes almost impossible. The authors go on to state that difficulties in debugging often arise from "*the programmer's lack of understanding of the robot's world view* [12]", and that augmented reality tools can address this by superimposing graphical representations of the robot's understanding of the environment on top of a live view of the environment itself [12]. Hence the programmer is able to see how the robot has interpreted the environment, and identify inconsistencies. The authors describe the image of the real world environment as the "*ground truth*" against which the robot's view can be compared and contrasted [12]. Figure 2 shows a visual example of this technique, where the data received from the robot's sonar sensors is converted to spatially situated 3D shapes and superimposed over the live image, and can therefore be verified visually by the user.

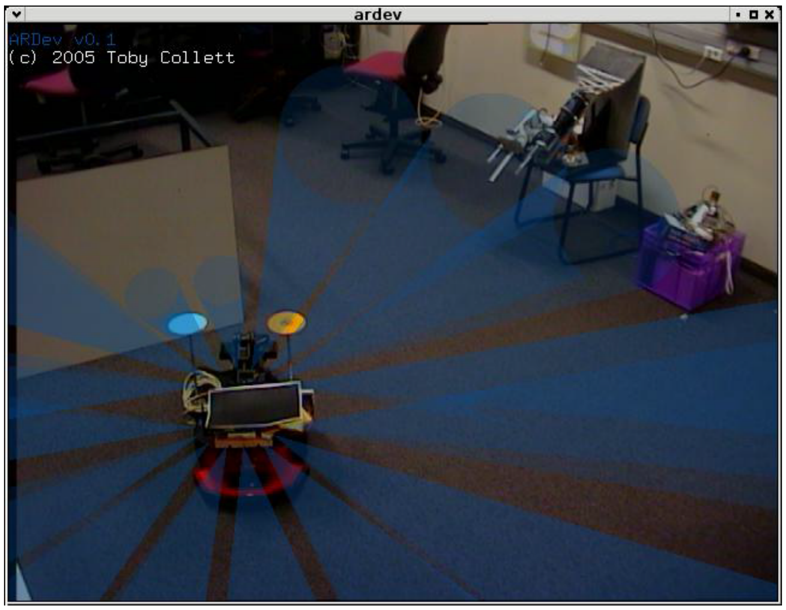


Figure 2: Sonar sensor data visualisation in Collet and MacDonald's system [12].

The application proposed by this project closely follows this paradigm; allowing the user to identify bugs by comparing the robot's knowledge of its environment and its decision making factors (collectively referred to as its state) with a view of the environment in real time. The application will aim to apply this concept specifically to swarm robotics systems, and therefore must allow the user to compare the states of multiple robots with the environment simultaneously. From the perspective of each robot in the swarm, the other robots will inherently form part of the environment, therefore the application must take this into account when displaying the information. Because of the large increase in information from a single-robot system to a multi-robot one, it becomes important that the application provide a way for the user to filter what information is displayed, allowing them to focus on the primary aspect under test. Being able to compare and contrast specific robots against one another by filtering out information related to other robots, or by displaying in more detail information related to the robots of interest, would also be a useful feature. Collett and MacDonald also present a separate paper [14] focusing on the creation of an AR visualisation plugin for the *Player* [13] robotics simulation environment, based on the principles in [12], which provides further insight into some relevant implementation details and practical considerations.

Ghiringhelli et al. [15] present a system for augmenting a video feed of an environment containing a number of robots with real time information obtained from each of the robots. This is similar in concept to the system described by Collet and MacDonald [12], but is designed specifically to target a multi-robot system. The authors identify the ability to overlay spatial information exposed by the robots on to the video feed in real time, in the form of situated graphical representations, as the most important debugging feature of the system. Figure 3 shows a spatially situated overlay of data exposed by robot thirty two, in the authors system [15]. A viewer is able to immediately verify the validity of the robot's world view from this image by comparing the blue overlay to the image beneath. Each robot features a coloured LED blinking a unique coded pattern to enable tracking, and the system uses homography techniques to map between the robots' frame of reference and the camera's. The project proposed in this report intends to use a simpler approach, with position and orientation tracking achieved through the use of the AruCo [16] marker-based tracking system, and a birds-eye view position for the camera to simplify mapping by effectively reducing the space to a 2D approximation.

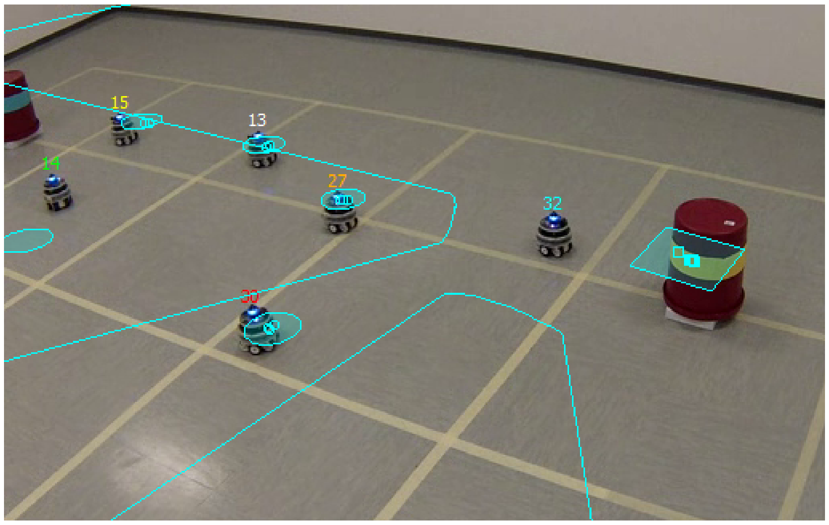


Figure 3: Example of spatially situated data overlayed on a live image in [15].

3 Project Objectives

With the idea of the project established in Section 1 and an understanding of the surrounding research area provided in Section 2, the overall aim and a set of objectives for the project can be summarised. The aim of the project is to understand the needs of a swarm robotics researcher or system developer when attempting to debug their system, and create a computer application which allows a user to monitor the state and behaviour of a robot swarm system in real time, thus improving the ease and efficiency of this debugging process. The objectives required to achieve this aim are as follows:

- Utilize existing fiducial marker based tracking technology to track the position of individual robots within a swarm over time.
- Develop code for presenting the user with a live video feed of a robot swarm, augmented with relevant and spatially situated information relating to the robots, using the data obtained from the tracking system.
- Develop code to allow multiple robots to communicate information regarding their internal state, sensor readings and decision making to the main application wirelessly via a network.
- Develop a data model that allows the application to store the information it receives from the robots, and update it as new information arrives.
- Develop code to ascertain higher level data related to the robots, such as recent movement history or state trasition history, and add this data to the model.
- Design and implement a user interface which presents the data model to the user in a human readable manner, and performs data fusion on the information provided by the robots and the tracking information.
- Develop the user interface in such a way as to allow the user to filter out information that is not currently relevant, and to contrast and compare information related to specific robots.
- Design and implement the system in a modular way so as to allow for relatively simple integration with other swarm robotic platforms and tracking systems in future extensions.

3.1 Software Specification

A commonly used technique when developing software is to define a specification prior to starting development in order to make clear the features the software must have and the things it must be capable of doing, in order to fulfil its purpose [17]. The following preliminary specification has been created for the software application proposed in this project. Any point within may be revised or removed during the course of the project should it be determined impossible or infeasible. The specification is divided into two groups; core requirements which are essential to the functionality of the software, and secondary requirements which will be implemented if possible in the available time frame.

Core Requirements:

1. Must be comprised of a PC application.
2. Must receive data related to the state of multiple robots.
3. Must receive positional data for the same set of robots.
4. Must receive a live video feed of the robots in their environment.
5. Must collate this data and present it to the user in a combined graphical form.
6. Must present auxiliary, non-spatial data to the user in textual or other forms.

7. Must update in approximately real time.

8. Must at minimum support the e-puck robot platform.

Secondary Requirements:

1. Should use a modularised structure.

2. Should exchange data between the robot platform and the application using a platform-agnostic, extensible protocol.

3. Should provide a basis for interoperability with a number of robotics platforms.

4. Should allow the user to configure the displayed data.

5. Should employ a model-view-controller (MVC) software architecture.

6. Could provide the user with ways to configure and display custom data types.

7. Could allow the user to compare data on two or more specific individual robots.

8. Could calculate and display swarm-level metadata and statistics.

9. Could generate log files of robot activity over a user defined period.

4 Project Plan

4.1 Timeframe and Approach

This project will take place between Monday 16th January and Thursday 18th May, 2017. In order to complete the necessary work outlined in Section 3 within this time, a Project Plan is presented. The work is divided approximately into logical tasks. Due to the nature of software development, the time required to complete a given development task is difficult to accurately predict. Hence any time lengths stated are best-guess estimates. Throughout the project an approach loosely based on 'Agile' development methodologies will be adopted [18]. Some Agile concepts such as *stand ups* are not relevant for a single developer project, however general concepts such as frequent requirements review, continuous parallel testing, and flexible task scheduling will be incorporated. Version control and issue tracking tools will be used to organise the development process.

4.2 Tasks

Table 1 gives a breakdown of the individual software development tasks required to complete the project. This list may not be exhaustive, as unforeseen or unexpected requirements may present themselves as the project continues.

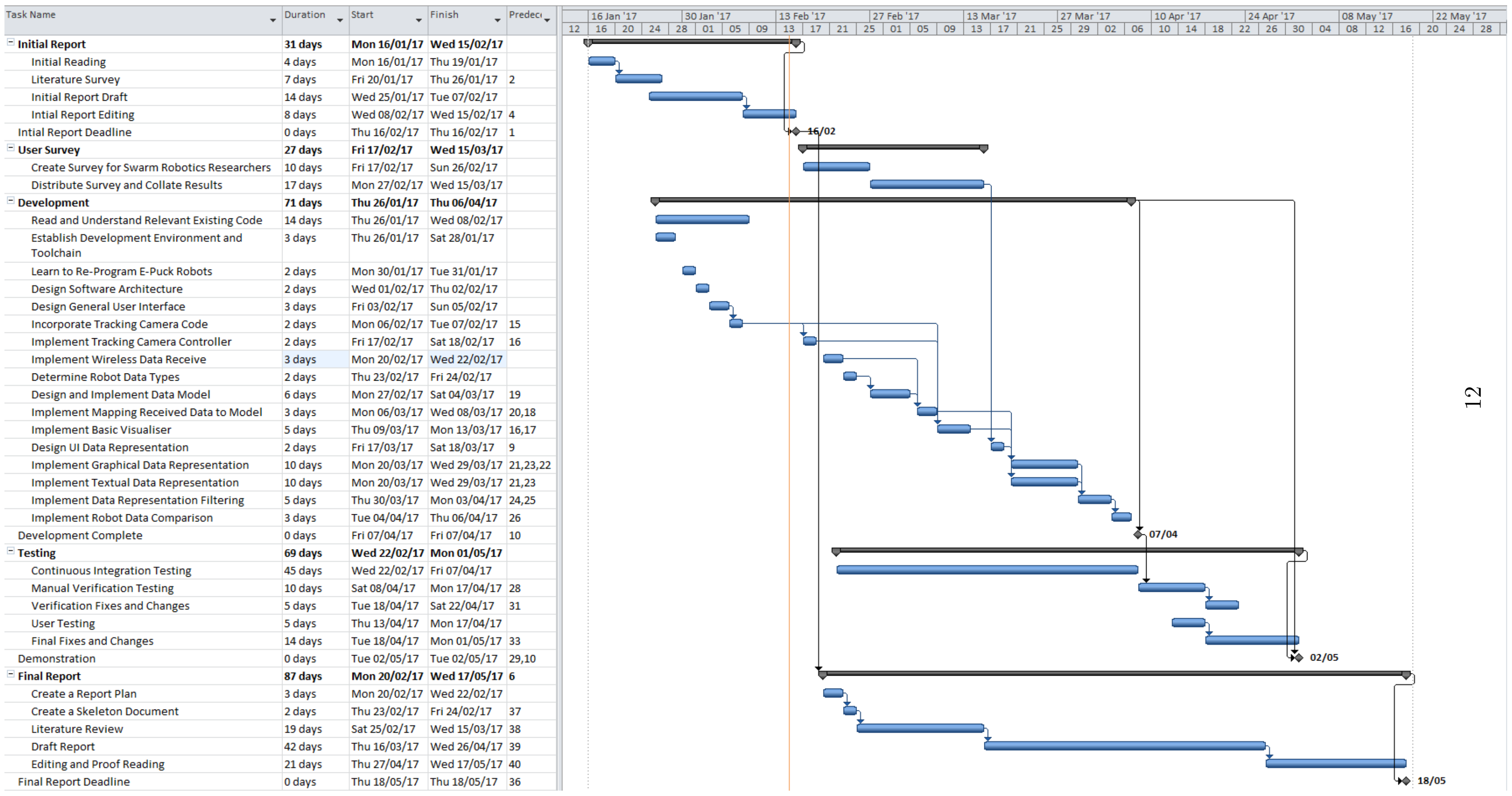
Task	Objective	Approximate Time
Read and Understand Existing Code	To understand existing code related to the tracking camera and networking on the e-pucks.	14 Days (Alongside other development)
Establish Development Environment and Toolchain	To enable organised development by establishing a tool set and workflow.	3 Days
Learn to Re-Program e-puck Robots	To understand the cross compilation process for the e-pucks.	2 Days
Outline Software Architecture	To design a coherent code structure in order for code to remain organised and modular.	2 Days
Design General User Interface	To create a high level design of the basic UI and implement a skeleton framework of this UI.	3 Days
Incorporate Tracking Camera Code	To incorporate existing low-level code for acquiring images from the tracking camera and performing tag detection.	2 Days
Implement Tracking Camera Controller	To implement code to create a layer of abstraction between the application code and the tracking code.	2 Days
Implement Wireless Data Receive	To implement code to allow the application to receive data wirelessly.	3 Days
Determine Robot Data Types	To establish an initial set of data types that will be supported by default, and a packet format for these.	2 Days
Design and Implement Data Model	To design the back end data model of the application and implement it in code.	6 Days
Implement Mapping Received Data to Model	To implement code to store received robot and tracking data in the application data model.	3 days
Implement Basic Visualiser	To implement code for displaying the video feed and augmenting it with basic geometric primitives.	5 Days
Design UI Data Representation	To establish a design for the representation of the different data types.	2 Days
Implement Graphical and Textual Data Visualisation	To implement code to convert the data in the data model into relevant visualisations.	10 Days
Implement Data Visualisation Filtering	To implement code to allow the user to filter out unnecessary visualisation elements.	5 Days
Implement Robot Data Comparison	To implement code to allow the user to compare the data of specific robots.	3 Days

Table 1: Development Tasks

4.3 Gantt Chart

Appendix A provides a Gantt Chart with the development tasks outlined in Section 4.2 shown in the context of the wider project. The chart includes a section on testing, which shows continuous integration testing running in parallel to development, as per the agile development mentioned previously. A section is also included for the creation of a user survey to be distributed to some potential users of the system. This will give insight into the application features they believe will be useful. The chart includes weekends as potential working days, but where possible development tasks have been scheduled to primarily fall on weekdays. As stated previously timings are approximate, and development tasks may incur slippage time due to unforeseen difficulties. The development completion date is therefore approximate, however the demonstration date is fixed, and development must be completed by this date. Slippage time will be accounted for where necessary by a reduction in the time allocated for testing, and the use of the currently uncommitted weekend days if necessary. The use of continuous integration testing should help to mitigate the risk associated with reduced testing. The final report will be written alongside the development work, with the aim of having the bulk of the report drafted prior to the demonstration date. This then leaves approximately three weeks for completing the report, including conclusion, editing and proof reading.

A Gantt Chart



Task Name	Duration	Start	Finish	Predecessors
Initial Report	**31 days**	**Mon 16/01/17**	**Wed 15/02/17**	
Initial Reading	4 days	Mon 16/01/17	Thu 19/01/17	
Literature Survey	7 days	Fri 20/01/17	Thu 26/01/17	2
Initial Report Draft	14 days	Wed 25/01/17	Tue 07/02/17	
Intial Report Editing	8 days	Wed 08/02/17	Wed 15/02/17	4
Intial Report Deadline	0 days	Thu 16/02/17	Thu 16/02/17	1
User Survey	**27 days**	**Fri 17/02/17**	**Wed 15/03/17**	
Create Survey for Swarm Robotics Researchers	10 days	Fri 17/02/17	Sun 26/02/17	
Distribute Survey and Collate Results	17 days	Mon 27/02/17	Wed 15/03/17	
Development	**71 days**	**Thu 26/01/17**	**Thu 06/04/17**	
Read and Understand Relevant Existing Code	14 days	Thu 26/01/17	Wed 08/02/17	
Establish Development Environment and Toolchain	3 days	Thu 26/01/17	Sat 28/01/17	
Learn to Re-Program E-Puck Robots	2 days	Mon 30/01/17	Tue 31/01/17	
Design Software Architecture	2 days	Wed 01/02/17	Thu 02/02/17	
Design General User Interface	3 days	Fri 03/02/17	Sun 05/02/17	
Incorporate Tracking Camera Code	2 days	Mon 06/02/17	Tue 07/02/17	15
Implement Tracking Camera Controller	2 days	Fri 17/02/17	Sat 18/02/17	16
Implement Wireless Data Receive	3 days	Mon 20/02/17	Wed 22/02/17	
Determine Robot Data Types	2 days	Thu 23/02/17	Fri 24/02/17	
Design and Implement Data Model	6 days	Mon 27/02/17	Sat 04/03/17	19
Implement Mapping Received Data to Model	3 days	Mon 06/03/17	Wed 08/03/17	20,18
Implement Basic Visualiser	5 days	Thu 09/03/17	Mon 13/03/17	16,17
Design UI Data Representation	2 days	Fri 17/03/17	Sat 18/03/17	9
Implement Graphical Data Representation	10 days	Mon 20/03/17	Wed 29/03/17	21,23,22
Implement Textual Data Representation	10 days	Mon 20/03/17	Wed 29/03/17	21,23
Implement Data Representation Filtering	5 days	Thu 30/03/17	Mon 03/04/17	24,25
Implement Robot Data Comparison	3 days	Tue 04/04/17	Thu 06/04/17	26
Development Complete	0 days	Fri 07/04/17	Fri 07/04/17	10
Testing	**69 days**	**Wed 22/02/17**	**Mon 01/05/17**	
Continuous Integration Testing	45 days	Wed 22/02/17	Fri 07/04/17	
Manual Verification Testing	10 days	Sat 08/04/17	Mon 17/04/17	28
Verification Fixes and Changes	5 days	Tue 18/04/17	Sat 22/04/17	31
User Testing	5 days	Thu 13/04/17	Mon 17/04/17	
Final Fixes and Changes	14 days	Tue 18/04/17	Mon 01/05/17	33
Demonstration	0 days	Tue 02/05/17	Tue 02/05/17	29,10
Final Report	**87 days**	**Mon 20/02/17**	**Wed 17/05/17**	**6**
Create a Report Plan	3 days	Mon 20/02/17	Wed 22/02/17	
Create a Skeleton Document	2 days	Thu 23/02/17	Fri 24/02/17	37
Literature Review	19 days	Sat 25/02/17	Wed 15/03/17	38
Draft Report	42 days	Thu 16/03/17	Wed 26/04/17	39
Editing and Proof Reading	21 days	Thu 27/04/17	Wed 17/05/17	40
Final Report Deadline	0 days	Thu 18/05/17	Thu 18/05/17	36

Figure 4: Project plan Gantt chart

References

[1] E. Sahin, "Swarm Robotics: From Sources of Inspiration to Domains of Application," in *Swarm Robotics WS 2004*, E. S¸ahin and W.M. Spears, Eds. Springer, Berlin, LNCS 3342, 2005, pp. 10–20.

[2] M. Dorigo and M. Birattari, "Swarm intelligence," *Scholarpedia*, vol. 2, no. 9, p. 1462, Sep. 2007. [Online]. Available: http://www.scholarpedia.org/article/Swarm_intelligence. Accessed: Feb. 5, 2017.

[3] F. Mondada et al., "The e-puck, a robot designed for education in engineering," in *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, vol. 1, IPCB: Instituto Politécnico de Castelo Branco, 2009, pp. 59–65.

[4] M. Dorigo et al., "Evolving self-organizing behaviors for a Swarm-Bot," *Autonomous Robots*, vol. 17, no. 2/3, pp. 223–245, 2004.

[5] S. Nouyan, "Path formation and goal search in swarm robotics," DEA thesis, Faculty of Applied Sciences, Universite Libre de Bruxelles, Brussels, 2004.

[6] G. Beni, "From Swarm Intelligence to Swarm Robotics," in *Swarm Robotics WS 2004*, E. S¸ahin and W.M. Spears, Eds. Springer, Berlin, LNCS 3342, 2005, pp. 1–9.

[7] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: From natural to artificial systems.* Oxford University Press, 1999.

[8] A. Kolling, P. Walker, N. Chakraborty, K. Sycara and M. Lewis, "Human Interaction With Robot Swarms: A Survey," in *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, 2016, pp. 9-26.

[9] A. Rule and J. Forlizzi, "Designing interfaces for multi-user, multi-robot systems," *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, 2012, pp. 97–104.

[10] R.E. Christ, "Research for evaluating visual display codes: An emphasis on colour coding," in *Information design: The design and evaluation of signs and printed materials*, R. Easterby and H. Zwaga, Eds. John Wiley and Sons, 1984, pp. 209-228.

[11] C. Carney, J. L. Campbell and E. A. Mitchell, *In vehicle display icons and other information elements: Literature review.* U.S. Department of Transportation, Federal Highway Administration, USA; 1998.

[12] T. H. J. Collett and B. A. MacDonald, "An augmented reality Debugging system for mobile robot software engineers," *Journal of Software Engineering for Robotics*, vol. 1, no. 1, 2010, pp. 18–32.

[13] B. P. Gerkey, R. T. Vaughan, A. Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems," *Proceedings of the 11th International Conference on Advanced Robotics*, 2003, pp. 317-323.

[14] T. H. J. Collett and B. A. MacDonald, "Augmented reality visualisation for player," *Proceedings 2006 IEEE International Conference on Robotics and Automation*, 2006, pp. 3954-3959.

[15] F. Ghiringhelli et al., "Interactive Augmented Reality for understanding and analyzing multi-robot systems," *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 1195-1201.

[16] S. Garrido-Jurado et al., "Automatic generation and Detection of Highly Reliable Fiducial Markers Under Occlusion", *Pattern Recognition*, vol. 47, no. 6, 2014, pp. 228 - 2292.

[17] NASA Software Engineering Laboratory, *Recommended Approach to Software Development*. National Aeronautics and Space Administration, MD, USA; 1992. [Online]. Available: http://homepages.inf.ed.ac.uk/dts/pm/Papers/nasa-approach.pdf. Accessed: Feb. 5, 2017.

[18] J. Highsmith and A. Cockburn, "Agile software development: the business of innovation," in *Computer*, vol. 34, no. 9, 2001, pp. 120-127.