1028: PICARSO

PROGRAMMABLE INTERFACE CONTROLLER WITH AUTONOMOUS ROBOTIC SPRAYING OPERATION

PRELIMINARY REPORT

21 May 2010

AUTHORS:
1147962 Ian HOOI
1146949 Samuel OOSTERHOLT
1148694 Sven PASCHBURG
1141447 Joyce PHAN
1142226 Neil YEOH

SUPERVISOR:
Ben CAZZOLATO
Executive Summary

The Programmable Interface Controller with Autonomous Robotic Spraying Operation (PICARSO) system is a cable-driven painting system, with image processing capabilities. The project aims to design and build a small scale prototype, nicknamed µAngelo, as an initial proof of concept, as well as an entirely functional full scale model, which will be referred to as PICARSO.

In the design development of PICARSO, it was advantageous to evaluate the successes and limitations of existing drawing robots and cable-driven manipulators, which contain elements of the concepts that will be employed by PICARSO. Most prominently, robotic painting systems such as Hektor (Lehni & Franke 2002), Viktor (Lehni & Rich 2008) and Rita (Lehni, Thurnherr & Acherknecht 2005) are at the forefront of robotic painting design in terms of overall system design sophistication and the remarkable accuracy and artistic expression of their output. Painting systems have also found commercial prominence in the popularity of the Robopainter businesses in the United States. Additionally, systems implementing cable-driven or tendon-simulating manipulators have been thoroughly researched.

The µAngelo prototype system has been developed as a small scale version of PICARSO without the painting capabilities. The testing of the µAngelo system allowed for improved understanding of three cable-driven parallel planar manipulator systems, and was able to demonstrate the feasibility of a similar system to be designed and built on a larger scale.

The full scale PICARSO will draw upon the successes of the prototype in the design development of the hardware, painting and control systems. Additionally, PICARSO will incorporate painting capabilities, image processing and motor feedback control to meet the design and functional specifications. Initially, the design will produce a monochromatic raster image and depending on time and budget constraints, added functionality may be implemented.

At present, the design and construction of the full scale PICARSO system is underway, with the specification, selection and purchase of commercial off-the-shelf components, as well as the design and drafting of custom components. Once manufactured and assembled, the image processing and control software will be implemented and tested. This process aims to develop a fully functional PICARSO system capable of producing accurate and high quality images.
Disclaimer

The content of this report is entirely the work of the five authors listed below. Any additional content obtained from other sources has been referenced accordingly.

......................................................
Ian HOOI
Date:

......................................................
Samuel OOSTERHOLT
Date:

......................................................
Sven PASCHBURG
Date:

......................................................
Joyce PHAN
Date:

......................................................
Neil YEOH
Date:
Acknowledgements

The authors of this report would like to express their sincere thanks to all persons who have helped with PICARSO to date. In particular, thanks to our project supervisor Associate Professor Dr Benjamin Cazzolato for his invaluable guidance and support.

We would like to acknowledge Phil Schmidt for his contribution to the electrical design through the provision of electronic components and knowledge. Special thanks to Jim Cormack from Anest Iwata for his continuous help in spray gun selection. Also, thanks to Dany Seif from APEX Automation and Robotics Pty. Ltd. who has donated a three-way solenoid valve. Additionally, we would like to acknowledge Maxon Motors Australia and in particular Brett Motom and Jon Pippard who have kindly offered an educational discount and their expertise for the selection of PICARSO’s motor system. Thanks, also, to Crowie’s Paints who have donated 30 litres of black paint, which will allow for over 100 painting sessions.

Finally, we would like to express our appreciation for The University of Adelaide 2010 Open Day Creativity and Innovation Fund for contributing towards the funding of this project.
The following section contains information regarding the prototype development.

B.1 Motor Specifications ................................................................. 90
B.2 PicoPic Specifications .............................................................. 91
B.3 Control Software Flow Diagrams .............................................. 92
B.4 Control Software Code ............................................................. 93
B.5 Design Calculations ................................................................. 100

C Mechanical System Custom Components 103

D Motors and Motor Components 111

E Spray Gun Components 117
   E.1 Air Brush and Spray Gun Options ........................................... 117
   E.2 Images of Various Types of Airbrushes ................................. 118
   E.3 Parts List for Anest Iwata SGA-101 Spray Gun ..................... 119
   E.4 Initial CAD Model for Spray Gun Design ............................. 120

F Image Processing 121
   F.1 Flowcharts for Image Processing ........................................... 121
   F.2 Image Processing MATLAB Code .......................................... 124

G Control Software 127
   G.1 Control Software Flowcharts .............................................. 127

H Risk Management 129
# Glossary

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D:</td>
<td>Two Dimensional</td>
</tr>
<tr>
<td>3D:</td>
<td>Three Dimensional</td>
</tr>
<tr>
<td>Binary Image:</td>
<td>A black and white image</td>
</tr>
<tr>
<td>CAD:</td>
<td>Computer Aided Design</td>
</tr>
<tr>
<td>CAN:</td>
<td>Controller Area Network</td>
</tr>
<tr>
<td>CP:</td>
<td>Centre Point</td>
</tr>
<tr>
<td>DC:</td>
<td>Direct Current</td>
</tr>
<tr>
<td>DoF:</td>
<td>Degree(s) of Freedom</td>
</tr>
<tr>
<td>DLL:</td>
<td>Dynamic-Link Library</td>
</tr>
<tr>
<td>EC Motor:</td>
<td>Electronically Commutated Motor</td>
</tr>
<tr>
<td>EPOS:</td>
<td>Easy-to-use Positioning (Motor controller series)</td>
</tr>
<tr>
<td>GUI:</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>I/O:</td>
<td>Input/Output</td>
</tr>
<tr>
<td>LARM:</td>
<td>Laboratory of Robotics and Mechatronics</td>
</tr>
<tr>
<td>Motor 1:</td>
<td>The central lower motor of the PICARSO and µAngelo systems</td>
</tr>
<tr>
<td>Motor 2:</td>
<td>The upper right motor of the PICARSO and µAngelo systems</td>
</tr>
<tr>
<td>Motor 3:</td>
<td>The upper left motor of the PICARSO and µAngelo systems</td>
</tr>
<tr>
<td>NC:</td>
<td>Normally Closed</td>
</tr>
<tr>
<td>NIST:</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>OH&amp;S:</td>
<td>Occupational Health and Safety</td>
</tr>
<tr>
<td>PC:</td>
<td>Polycarbonate</td>
</tr>
<tr>
<td>PICARSO:</td>
<td>Programmable Interface Controller with Robotic Spraying Operation</td>
</tr>
<tr>
<td>ProE:</td>
<td>Pro-Engineer Wildfire 5.0 (CAD software package)</td>
</tr>
<tr>
<td>PVC:</td>
<td>Polyvinyl Chloride</td>
</tr>
<tr>
<td>RC:</td>
<td>Remote Controlled</td>
</tr>
<tr>
<td>RGB:</td>
<td>Red Green Blue (Image format)</td>
</tr>
<tr>
<td>RS232:</td>
<td>A serial communication interface for control signals</td>
</tr>
<tr>
<td>SEGESTA:</td>
<td>Seilgetriebene Stewart–Plattformen in Theorie und Anwendung</td>
</tr>
<tr>
<td>USB:</td>
<td>Universal Serial Bus, a serial communication interface between devices and a host controller</td>
</tr>
<tr>
<td>Venturi Effect:</td>
<td>The reduction in fluid pressure that results when fluid is constricted through a section of pipe (Spiritus-Temporis 2005)</td>
</tr>
</tbody>
</table>
1 Introduction

The Programmable Interface Controller with Autonomous Robotic Spraying Operation (PICARSO) system is a three-cable-driven parallel planar manipulator with image processing capabilities. The system will be scalable and vertically oriented, with a spray painting tool housed in the end-effector. This project aims to design and build two versions of PICARSO, including a simplified scaled down system without spraying operation and a full scale system with autonomous spraying capabilities. The scaled system will be referred to by its nickname 'μAngelo', to distinguish it from the full scale system, which will be referred to as 'PICARSO'.

The PICARSO project is a systems engineering project and thus will include a literature review of existing painting robots and cable-driven parallel manipulator systems, the design and build of the two PICARSO systems, the implementation of image processing capabilities and low to high level control. The initial PICARSO system has been designed to output a monochromatic raster image. If time and budget permit and the core objectives are achieved, additional functionality such as painting via a vector based approach and greyscale painting will be implemented.

The full scale system will involve the design of the mechanical system, which includes motors and cable configuration; the control system, which includes the control electronics hardware and software; and the painting system, which includes the painting module, compressor and paint. Additionally, full scale testing of the system will be undertaken and the results evaluated against the project goals.

1.1 Background

Several autonomous painting systems have previously been built, which contain elements of the concepts that will be utilised in PICARSO. Articles have been published on cable-driven parallel manipulators, with some investigation into the planar operation of these manipulators.

Hektor, is an example of an existing autonomous painting system, documented by Lehni and Franke (2002). Hektor is a portable, two cable-driven spraying painting system that is capable of painting vector images. RoboPainter (2008) operates a robotic system utilising digital airbrush technology to paint decorations and advertisements onto almost any surface. This system provides an example of scalable robotic spraying operation as well as displaying the potential commercial viability of robotic painting systems.
Additionally, studies have been undertaken into the implementation of cable-driven parallel manipulators, which form the basis for the structural configuration and design of the mechanical system of PICARSO. One investigation was performed by Ottaviano, Ceccarelli and Pelagalli (2006) from the University of Cassino, Italy, and analysed the performance of a four cable-driven parallel manipulator, which will be further discussed in Section 2 and was used as a reference for the mechanical system design.

1.2 Scope and Objectives

This project will focus on the design and build of two systems, PICARSO and µAngelo. The µAngelo system will be limited to the design and build of a cable-driven parallel planar manipulator system, which is effectively a scaled down version of PICARSO without painting capabilities. PICARSO, on the other hand, will be a full scale cable-driven parallel planar manipulator system with automated painting capabilities. PICARSO will initially be limited to monochromatic raster based painting with image processing functionality, however time and budget permitting, PICARSO’s capabilities may be extended to possess vector based or greyscale painting ability. The core project objectives are detailed below.

1.2.1 Core Objectives

This section details the core project objectives. The success of the project will depend on the achievement of these objectives.

Development of Image Processing Capability

An independent image processing program will be developed which will process any standard picture file readable by MATLAB as an input and produce a monochromatic image in a raster format. The image processing program will then be integrated with PICARSO, which will send the processed image in a format that can be converted into a set of instructions for the motors and painting mechanism. The program must use appropriate filtering and transformations to obtain an image of reasonable quality once painted. The image processing program is likely to be modified through an iterative process to suit the requirements of the other parts of the design and should be successfully integrated with the main controller system via Simulink, or otherwise, at the assembly stage.

Design of a Scaled System - µAngelo

A simplified three cable-driven parallel planar manipulator system will be designed and built. The aim of µAngelo is to allow for a better understanding of cable-driven parallel planar
manipulators, specifically the operation and kinematics of such systems. Furthermore, µAngelo will be used to determine any potential limitations of these types of systems and to evaluate any possible implications this has on the full scale system.

**Design of a Full Scale System - PICARSO**

Following evaluation of µAngelo, a full scale, three cable-driven parallel planar manipulator will be designed and built. The PICARSO system will incorporate all of the primary design specifications. Extension components will be incorporated into the system upon the successful completion of the initial system design and achievement of the core project objectives.

**Lower Level Control**

A program will be developed which will be responsible for the lower level control of the system. These levels of control will be capable of converting the information of a processed image provided into a set of instructions that can direct the motors. The motor sequence will be such that, the resultant output may be a scaled reproduction of the input image. The program will direct the motors to a sequence of positions and actuators will control the painting mechanism to produce the desired image by using a raster based approach.

**Higher Level Control**

A program responsible for the higher level control of the system will also be developed. Higher level control will be developed to incorporate a graphical user interface (GUI) for image input, image processing and starting the autonomous painting of the image. This will allow for more user friendly control in the final system.

**1.2.2 Extension Goals**

Additionally, if time and budget permits, this project aims to add extra functionality to PICARSO through the extension goals. It must be noted that the achievement of these extension goals depend on the success of the core project goals detailed above.

**Vector Based Painting**

The project may be extended by incorporating a vector based painting system. In this case, PICARSO will outline an image as a series of lines and curves rather than by pixels. Vector based imaging means that the control software for the system will need to be rewritten for this alternative painting approach.
**Greyscale and Colour Painting Capability**

As an additional goal and upon the successful completion of the monochromatic design, the image processing program will be modified to take into account different intensities and colour tones. The subsequent output of the image file will be in a greyscale or colour format. This functionality may involve the modification of the painting module, especially in the case where colour painting capability is implemented.

**Implementation of Added Functionality**

PICARSO will include the following added functionalities, upon the successful completion of the core project objectives, so the system can be made more user-friendly. These goals are resource and time dependent.

**Wireless**

As an extension goal, PICARSO will be controlled wirelessly. The removal of physical connections will improve portability of the system. Additionally, batteries will also be implemented to remove the connection to mains power.

**Touch Screen**

Depending on available funding and time constraints, a touch screen interface may be used as an interactive input device. The inclusion of a touch screen interface will allow users to trace a desired image, which will then be replicated by PICARSO.

At the time this report was written, it must be noted to that not all core project objectives have been achieved, however the project is making progress with respect to the project Gantt chart (see Appendix A). The design development of the prototype, µAngelo can be found in Section 1. While a progress report of PICARSO for each sub-system can be found in Section 1. A discussion of the remaining work required for the successful completion of the project can be found in Future Work (see Section 1).
2 Literature Review

The literature review detailed in this section was conducted to ascertain and understand the successes and failures of painting or drawing systems that have been designed before. In attempt to gain a better understanding for the design development of PICARSO, each existing system was analysed, where applicable, in terms of PICARSO’s separate design areas: cable drives, motor type and control, image processing techniques and the method of image output. Additionally, cable-driven manipulator systems have also been evaluated for the derivation of the most appropriate design configuration for implementation to this project.

2.1 Hektor

Hektor was a design project of Jurg Lehni and Uli Franke, who in 2002, designed a cable-driven painting mechanism which used a dual motor, cable-rigged configuration to drive a mechanically actuated spray can. Hektor was capable of processing a monochromatic image by determining the vector pathways which subsequently drove the spray can to the desired positions. The accuracy of the image that Hektor was capable of producing was remarkable and provided motivation in the development of PICARSO as shown in Figure 2.1.

![Figure 2.1: A raster based image produced by Hektor (Lehni and Franke 2002, p. 45).](image-url)
The Hektor drive system initially implemented four stepper motors to function as winches in an X-configuration. The utilisation of stepper motors was mainly due to efficiency, low cost and simplicity in control, which was acceptable due to the early stage of the prototype’s development as a proof of concept design. Lehni (2002, pp. 10-11) justified the use of two motors by discussing that, “the lower [two] motors caused instabilities and resonant frequencies and therefore had to be removed.” Thus, in their design process, the lower two motors were removed, leaving only two motors positioned in the upper left and right corners of the workspace. The final design for Hektor incorporated a cable-driven manipulator that used two tooth belts to guide its end-effector in the vertical x-y plane.

Hektor’s painting system consisted of a standard spray can in a metal housing, operated by a programmable actuator, and used to produce images. The choice of painting with a mechanically actuated spray can was due to the desire to create a means for artistic expression rather than a display of precision in replicating an image. Hektor’s painting system design prompted a range of ideas and research pathways. Their idea of producing images using projectile paint was innovative, and reduced problems that may occur from contact to the painting surface, for example friction. Also, it allowed more flexibility in terms of the distance which the paint could be sprayed from, and increased the variety of paintable surfaces. Commands to the painting system were written using Scriptographer, which was a scripting plugin created by Jurg Lehni for the purpose of taking the vector image files from Illustrator and converting them into instructions for the motors. Furthermore, the plugin was designed to communicate through a serial interface, which was considered as a suitable choice for PICARSO. Overall, this method allowed sufficient control and image processing capabilities to successfully produce both raster and vector images.

There are advantages of the Hektor system which can be considered for the PICARSO system. For instance, Hektor designed their painting system to be a separate component which eliminated any external obstructions, such as connecting tubes and wires which may affect the painting module’s movements. Moreover, the choice of spray cans for the painting module was simple, cheap and allowed the successful production of both raster and vectored images from Scriptographer commands. These commands allowed effective control of the image processing. Additionally, Illustrator was able to define and convert an image’s vector pathway into motor commands to drive Hektor. This allowed the development of a program that could express vector pathways through a set of motor commands by finding the algorithms required to make those conversions. In addition, the implementation of projectile paint to produce images on the
wall was an important design selection, which motivated research into a suitable painting module design that increased painting accuracy and minimal hindrances during movement.

Conversely, the objective to construct Hektor to capture the imperfections of art, in a similar way to the production of human art, was in contrast to the design objectives of PICARSO. Subsequently, there were several design issues found within this system which were considered in the PICARSO system, for instance, weight changes due to paint expelled from the spray can, non-instantaneous and inconsistent paint distribution (Figure 2.2), dripping of paint once sprayed onto the canvas and lack of control from a single actuated nozzle.

![Figure 2.2: Image illustrating inconsistent paint distribution in Hektor (Lehni and Franke 2002, p. 29).](image)

Also, the two-motor configuration required a slower running speed for smooth operation due to the small mass of the spray can end-effector which in turn, increased the complexity of the computer code. In terms of motor configuration, there are numerous improvements that could be achieved to find a more suitable drive system for PICARSO. Despite understanding the merit and intentions of Hektor’s development as a work of art, the aims of PICARSO were based on quantifiable performance outcomes. This largely affected the design options considered for the PICARSO system.

### 2.2 Viktor

The Viktor system was a vertically mounted image producing robot designed by Jurg Lehni and Alex Rich in 2008. This system, which was an evolution of Hektor, returned to the initial four motor design. This time, Festo servo drives were implemented instead of stepper motors. The decision to change the motors from the stepper motors used in the predecessor, to servo motors, was dependent on the increase in complexity of the end-effector and cable rigging system. Viktor implemented four cables to guide its end-effector in the vertical x-y plane. The
wall mounting system comprised of modular units each housing a single motor, motor controller and spooling system. The image processing software was retained from the Hektor robot but modified for the control of four servo motors instead of two stepper motors. In the development of Viktor, the creators decided to eliminate the use of sprays cans, and instead chose to draw the desired image using chalk on a blackboard surface. With the increased weight of the painting system, higher torques and speeds needed to be considered to allow for accurate movement. As the functional requirements of Viktor became more complex, its hardware needed to be upgraded. This was one of the major considerations when choosing the motor type for the PICARSO system.

There are advantages of the Viktor system which can be considered for the PICARSO system. For example, the number of motors used in the system had increased from two to four for increased stability and manoeuvrability across the canvas. Considering this design direction, the PICARSO system would employ a minimum of two motors above the working space, and incorporate at least one motor below to obtain the benefits of increased stability. Moreover, Viktor’s use of servo motors allowed for a much smoother motion and much higher resolution for position control. Using servo motors for the PICARSO system would provide the same ability as Viktor for feedback control and correction of errors, without the implementation of additional electronic components. Viktor’s compact motor mount design is desirable for the PICARSO system to meet portability and scalability objectives. Furthermore, the axis of rotation of each motor on Viktor was perpendicular to the plane of the end-effector, allowing the spool to attach directly to the shaft of the motor, thus restricting the cables to a singular plane. This was a simple and economic solution to spooling the cables, which was considered for PICARSO.

Conversely, Viktor’s implementation of a piece of chalk as the tool on its end-effector, like Hektor’s use of a spray can, was chosen for artistic expression rather than a precise method of image production. Additionally, Viktor’s solution to spooling the cables, resulted in a significant protrusion of each motor from the wall, which was less aesthetically pleasing than having the motors attached parallel to the wall. Despite these disadvantages, the Viktor system was able to reproduce chalked images with good accuracy due to the robustness of its image processing and path describing command software. Subsequently, much was learnt from Viktor’s robust control, motor configuration and design, and thus considered for the PICARSO system.
2.3 Portrayer

The Portrayer robot was a drawing arm designed by Daniel Benedettelli in 2008. The Portrayer was driven using the Lego Mindstorms NXT system and controlled using the packaged controller. The Lego package was used due to the elegance of the packaged system and software, and the minimal, but adequate torque requirements of the motors to support a single drawing implement.

This system used the NXT Lego Controller to control a pen held in a robotic arm which sketched the desired image in vector form, as seen in Figure 2.3. The vector algorithm and all the image processing capabilities were written in MATLAB. The path finding algorithm, however, was based on another system, Erik’s X-Y plotter (Convict Episcopal de Luxembourg 2007). The algorithm converted points from a raster image into chains of co-ordinates which represented a vector. Using this process, lines could be defined using coordinate points and hence the actual equations of the lines would not be required. This simplified the conversion from raster to vector form. The MATLAB program was ported to the NXT using an MEX file, which was used to interface between C++ and MATLAB (Mathworks 2010). It was thought that a similar approach could be used if PICARSO was to implement a vector based approach.

![Figure 2.3: The Lego Mindstorms based Portrayer robot (Benedettelli 2008).](image)

The most obvious difference between Benedettelli’s Portrayer robot and the objectives of PICARSO, was the implementation of a robotic arm to drive the end-effector rather than using a cable-driven system. Due to this difference, Portrayer's mechanical design was not influential in
PICARSO's final design. However, the reason for considering this system in depth was due to the success of the image processing and the vector pathways software.

2.4 Robopainter and Rita

Robopainter, a commercial robotic airbrushing painting system service provided in the United States, and Rita, an art installation (Lehni et al. 2005), have been used for vastly different applications but both employ similar mechanical systems to that of a traditional x-y plotter. These plotter type systems drive its end-effector to a desired position by describing x and y coordinate motion along two fixed parallel bars and a moving perpendicular bar. The tool on Robopainter consisted of four coloured spray guns, while Rita's tool used different coloured whiteboard markers to produce coloured images.

As a commercial system, exact specifications of Robopainter's hardware and software were not available for public access and therefore, could not be evaluated. However, the primary concept taken from the system design was the choice to use spray guns to deliver paint to the canvas as shown in Figure 2.4.

![Image](image.png)

Figure 2.4: An example of an image painted by RoboPainter Texas (RoboPainter 2008).

The painting mechanism of Robopainter produced accurate raster images onto a number of surfaces. For the painting system, design considerations such as a supporting rigid horizontal bar, or the utilisation of sensors, and actuators to control the distance between the surface and the spray gun, were considered. The sensors allowed for accurate and consistent spray over the surface. Moreover, the supporting horizontal bar for the painting module allowed for accurate horizontal movement and produced accurate pixel positions. However, applying these features required more intricate mechanical design and resources, hence restricted it as a possibility for the scalable system.
The Robopainter system is an elegant design and effectively produces images onto a vertical canvas. The overall system, including the large volume of machinery, automation and spray guns, as well as the paint, materials and resources, operates efficiently, however developing PICARSO based on this design concept would require too much time, labour and cost.

2.5 Cable-Driven Parallel Manipulators

Cable-driven manipulation is often employed for end-effector positioning in robotic systems. Research into cable-driven manipulators is important due to their relevance to various real world applications, both in manufacturing and medicine. This area of research is populated by numerous systems developed by educational institutions, some of which are evaluated in this section.

LARM Four Cable-Driven Parallel Planar Manipulators

The Laboratory of Robotics and Mechatronics (LARM) developed a prototype using four cable-driven parallel planar manipulator consists of a platform end-effector (Ottaviano 2007). The end-effector was manoeuvred by cables controlled by a DC motor attached to a cubic frame. The LARM team developed the manipulator so that to be used in both planar (2D) and spatial (3D) applications, which resulted in utilising four sets of cables and motors to control the end-effector and to allow for flexibility in its constraints and orientation when moving between applications, as seen in Figure 2.5.

![Figure 2.5: A four cable-driven parallel planar manipulator (Ottaviano et al. 2006, p. 2).](image)

LARM implemented DC motors for cable-actuation and to drive the end-effector. The decision to employ DC motors, was due to cost efficiency and the simplicity of controls used to drive the
motors. The motors were Pittman GM 941, which were chosen as they were easily accessible and where able to achieve the required torque and speed for the design.

The prototype was a spatial parallel manipulator operating across a horizontal plane. The end-effector was suspended by cables with pulleys located on a rigid frame. This configuration enabled the end-effector to reach positions in 3D space at multiple angular orientations, refer to Figure 2.6, but depended only on gravity for the downward force component. From the perspective of the project objectives, the PICARSO system did not require control beyond the plane of the desired painting surface, thus rendered two of the motors redundant. A notable design feature was the use of simple pulley system which allowed the motors to be mounted and spooled from the base of the frame.

LARM's cable driven manipulator was used to gain an understanding for the calculations of the PICARSO system's inverse kinematics, forward kinematics and torque requirements. Another advantage to understanding the LARM manipulator system was its use of cables to drive its end-effector and the decision making process to determining the exact number of cables that would be suitable for the system.

LARM's cable driven manipulator was used to gain an understanding for the calculations of the PICARSO system's inverse kinematics, forward kinematics and torque requirements. Another advantage to understanding the LARM manipulator system was its use of cables to drive its end-effector and the decision making process to determining the exact number of cables that would be suitable for the system.

The Seilgetriebene Stewart–Plattformen in Theorie und Anwendung (SEGESTA) prototype was a testing rig by Hiller, Fang, Mielezarek, Verhoeven, and Franitza (2004) that operated in 3D space. The system was used to test different configurations of tendon-based manipulators with the ability to employ up to seven cables for six degree of freedom (DoF) control. The end-effector was a planar, triangular plate with two cables at each of two apices and three cables fixed at one apex, see Figure 2.7a and Figure 2.7b. Like the previously discussed LARM cable-driven
manipulator, the SEGESTA rig was over specified with regards to the PICARSO system, as 6 degree of freedom control was too complex and a design of that level of difficulty was not required to achieve PICARSO’s project objectives. However, employing additional cables would allow control over pitch and yaw of PICARSO’s painting mechanism. Other noted design features of SEGESTA that the PICARSO system would benefit from, included the use of pulleys to allow the motors to be mounted away from the testing area, and a ‘ceramic eye’ to provide a defined position of the end of the cable, as seen in Figure 2.7c.

![Components of the SEGESTA prototype. (a) Platform. (b) Manipulator. (c) Winding system configuration (Hiller, Fang, Mielczarek, Verhoeven, and Franitza 2004).](image)

Similarly, the National Institute of Standards and Technology (NIST) 6m high RoboCrane by Albus, Bostelman and Dagalakis (1992) and its prototype successor by Riehl, Gouttefarde, Krut, Baradat and Pierrot (2009) were both cable supported platforms with 6 degrees of freedom. The systems used elevated pulleys to guide two lines to the vertices of a triangular end-effector, which allowed control of each of the triangle’s sides. Additionally, high torque motors were located at the base of the solid support frames. Again, the use of two cables was a design feature that allows for 6 degree of freedom control, but could be applied to the PICARSO system to control yaw and pitch if required.
3 Project Specification

This section details the project specifications, including the design and functional specifications for each sub-system and additional constraints on the overall system. PICARSO has been broken down into sub-systems, consisting of the mechanical system, which incorporates the painting system and the image processing and control system, as illustrated in the flow diagram of the system architecture in Figure 3.1.

Figure 3.1 PICARSO system architecture.
3.1 Mechanical System

This sub-section details all specifications for the mechanical system in both PICARSO and µAngelo. The mechanical system of µAngelo consists of the hardware components of the system, while the mechanical system of PICARSO consists of all mechanical and electronic components of the system.

3.1.1 µAngelo System Design

The design specifications for the µAngelo system are as follows.

- The expected printing area of the scaled system is 0.3×0.3m in a vertical plane.
- A painting mechanism will not be included in the end-effector, however, this may be modified to incorporate a writing implement to demonstrate the outcomes of the systems based on results.
- The microcontroller processor will be a PicoPic if an operating unit can be sourced.

3.1.2 PICARSO Mechanical System Design Requirements

This PICARSO mechanical system design specifications are detailed below.

- The full scale system will be capable of producing an image within a vertical area of up to 3.0×3.0m, and will also aim to achieve the extension goal of producing an image within a vertical area of up to 5.0×5.0m.
- A cable driven manipulator system is required to guide and move the end-effector module along a desired path.
- A three motor, ‘Y’ configuration will be employed.
- The cables must be capable of suspending a force of 65N and wound around a spool of approximately 0.1m diameter.
- The system must consist of an appropriate solution for wall mounting.
- The system shall include motors capable of driving the painting module to the desired positions.
- The motor system will consist of the motor, gear head, encoder and motor controller.
- The motors shall be capable of a minimum torque of 3Nm.
- For initial design purposes, the motors shall be capable of operating at 150rpm, but ultimately, this speed can vary.
- The system shall include a controller board for the high level programming control of the system.
3.1.3 PICARSO Mechanical System Functional Requirements

The PICARSO mechanical system functional specifications are detailed below.

- The system shall be capable of adapting to a change in the height and/or width of the spraying area (i.e. scalable).
- The cable driven manipulation system, actuators and end-effector must be easily transportable.
- Consideration must be made for kick back of the gun due to the exiting pressurised air.
- The system design shall incorporate the housing of motors, motor controllers and spooling system when wall mounted.
- The motors must allow the system to operate as effectively, over a small scale as to a large scale, to allow for scalability.
- The motors shall be able to be controlled by power amplifiers which shall allow for feedback control.
- The controller shall have the ability to integrate with the MATLAB and Simulink programs, which will be used to program the controller and software.
- The controller shall be able to interface with a desktop computer.

3.2 Painting System

This subsection details all specifications for the painting system of PICARSO, which consist of all components required to provide painting functionality to the system.

3.2.1 Painting System Design Requirements

The PICARSO painting system must incorporate the following design specifications.

- The painting system shall consist of three main components: the painting module, the compressor and the paint. An additional three-way solenoid will be a part of the control system.
- The painting module shall remain separate from the compressor and paint components in order to keep the weight of the module constant and minimise the changes to the dynamics of the painting system.

Painting Module

This section details the specifications relating specifically to the painting module of the system.

- The painting module shall be capable of painting the input picture onto a vertical surface.
- The surface shall be flat with no inconsistencies (e.g. bumps and cracks).
• The surface shall be suitable to be drilled into for the mounting of the system.
• The painting module shall be guided by the mechanical system as specified in Section 3.1.2.
• The painting module shall further divided into spray gun, and other required components, and the housing to encase these components.
• The spray gun (without paint) shall be less than 1.0kg.
• The housing shall weigh no more than 2.0kg.
• The size of the housing shall be smaller than a 0.5x0.5x0.5m cube. However, be of a sufficient size to encase the spray gun and other required components.
• The housing for the painting module shall be made out of a material that will meet the weight specification.
• A spray gun shall be chosen as the paint spraying mechanism.
• The spray gun shall be reliable, durable, and be able to repeatably produce images of good quality.
• The spray gun shall have a suitable spray area size of 3.0x3.0 metres.
• The spray gun shall have a nozzle that can produce a circular pattern of a diameter between 0.01 to 0.05 metres.
• The spray gun shall have a suitable triggering mechanism that can be modified to fit the system specifications.
• The fire rate that the trigger is capable of shall be greater than 1Hz.
• A three-way solenoid valve shall be used external to the painting module to activate the spray gun.
• The frequency capability of these actuators must be greater than 1Hz.

Compressor

This section details the specifications relating specifically to the compressor.
• A compressor shall be used to feed pressurised paint into the painting module.
• The pressure range of the compressor shall be between 20psi – 80psi of constant air pressure.
• The compressor shall be located separate to the painting module.
• Tubing from the paint pressurised container will feed the pressurised paint to the painting module.
• The length of the tubing shall be less than 3m.
Notes:

Several details must be noted with regards to the painting system design specifications, above. These include:

- certain limitations on the capabilities of the spray gun that must be taken into account. These limitations include the gun having a kick back due to the pressurised air, as well as the limitation on the spray area, both of which may decrease the accuracy of the system and quality of the painted image;
- considerations such as the length of the tubing, gravity, and paint quantity must be accounted for when choosing the appropriate compressor to ensure that the paint exits the paint module at a suitable rate and pressure;
- minimal cost, but high performance paint would be most suitable for the painting module. The paint can be housed in a pressurised container located on the ground and connected to the painting module through tubing. If the paint were to be located on the module itself, problems will be faced due to weight changes, changes in module dynamics, and stresses on the motors;
- the paint in the spray gun will be atomised when released. The majority of the paint will attach itself to the canvas but a certain percentage will be released into the air. This means that the spray gun must be operated in a well ventilated area with the correct safety equipment worn (e.g. gas/vapour filter masks) to abide with OH&S requirements.

3.2.2 Painting System Functional Requirements

The PICARSO painting system must incorporate the following functional specifications.

Paint

This section details the functional specifications relevant to the paint.

- Paint is required to produce the image on the canvas.
- The paint shall be chosen to minimise the clog in the spray gun. Moreover, it shall be chosen to minimise bleed or drip. Also, the paint shall dry at a suitable rate for the raster imaging speed.
- The paint shall be housed in a pressurised container located on the ground and connected to the painting module through tubing.
- Paint quantity in the pressured paint container shall be enough for a single uninterrupted paint session.
3.3 Control System

This sub-section details all specifications for the control system of PICARSO, which consists of the image processing software and the control software.

3.3.1 Image Processing Software

The image processing software specifications are detailed below.

- Image processing functionality for the system shall be programmed using MATLAB and the inbuilt Image Processing Toolbox. If a vector approach is attempted, an external program may be used such as Adobe Illustrator or an equivalent software program.
- The system shall be able to process any MATLAB compatible picture formats, including JPEG, GIF, BMP and TIFF as stated in the MATLAB help file (MathWorks 2001).
- The picture will have a file size of no more than 10Mb in order to reduce processing time.
- The initial program shall be able to process images in raster form and filter and transform these images into either matrix form, a set of coordinates or otherwise, in order to be read by the main controller.
- The system's initial design shall be able to convert the input image into a monochromatic or binary form.
- Depending on the capabilities of the spray gun, the program shall be able to convert the input image into a greyscale form, and scale the gradients to suit the spray mechanism’s capabilities.
- The program will be able to complete the image processing and conversion within 60 seconds of starting the program depending on large input file sizes.
- Resources permitting, a further extension of the project will incorporate vector based instructions to be sent to the controller. The program will convert the raster input image to a vector form as a series of mathematical curves or otherwise. It is desirable to only include major edges, such as outlines from the picture, while ignoring shaded areas.

3.3.2 Control Software

The control software will incorporate the following specifications.

- The system shall include three levels of control: high, intermediate and low levels.
- The responsibilities for each level of control will be developed as equipment is chosen and the limitations become apparent.
• The control levels will include functionality such as a graphical user interface (GUI) which will allow higher level user operational control, position control of the end-effector, spooling of motors, as well as actuator triggering.

• The system shall include the implementation of the control software for the purpose of controlling the motors, sensors and spray gun trigger actuator.

• Where possible the control software programming shall be executed in MATLAB and Simulink programs.

• The control software shall be capable of running image processing on an image in raster format and convert it into a file containing a consecutive sequence of pixels either requiring paint or not, when using a raster approach.

• The control program shall be able to call this file and convert this sequence into the low level control of the motors (i.e. winding and unwinding the spools to direct the painting module to the desired position).

• The control program shall be able to control the trigger actuators for the activation of the spray gun.

• As an extension, the control program shall be able to perform the aforementioned tasks using a vector based approach.

3.4 Additional Constraints

Additional design constraints and considerations that must be taken into account in the system design are detailed below.

• The budget constraints on the system will be dictated by the ability of the project to attract suitable industrial sponsorships or donations.

• The initial budget constraint for the system is $200 per student, which equates to $1000 for the team.

• The full scale system shall be capable of painting an image, 3×3m in size, within 1 hour.

• The extension design will use a wireless communication interface and batteries capable of powering each of the actuators or the end-effector.

• The controller shall be capable of reading and writing a sufficient number of input and output ports for control of motors, sensors and paint spray gun actuators.
4 Prototype Development - µAngelo

The following section presents the design development of the PICARSO prototype, µAngelo, including the evaluation of its testing. As a part of the evaluation process, the final design and system limitations will be discussed for each of µAngelo’s sub-systems. The evaluation and performance of µAngelo will significantly influence the design development of the final PICARSO system, therefore the analysis of the outcomes are vital.

There were three main purposes of the design and build of the µAngelo system. Firstly, the system led to a better understanding of cable-driven parallel planar manipulator systems through testing and evaluation. Secondly, the system acted as a prototype test bed for the kinematics of a cable-driven manipulator, and also as a proof of concept for the implementation of a three motor configuration. Thirdly, the system limitations were determined through testing and evaluation, which may then be applied to PICARSO.

µAngelo was designed to use relatively inexpensive and readily available components due to time and budget constraints. In order to complete this system as soon as possible, the design and manufacturing process was simplified. The details of this process can be found in the following sections.

4.1 Hardware Design

The hardware design of µAngelo consisted of five main components including motors, cables, the motor controller, the end-effector and the mounting board. As per the project specifications, µAngelo was designed to operate within a workspace of at least a 0.3×0.3m area, and therefore system components were chosen on this basis as well as cost efficiency and availability.

4.1.1 Motor Selection

The main aim of the simplified µAngelo system was to analyse position control of the cable-driven manipulator system. The objective of such a system was to place the end-effector at a desired location as accurately as possible. Consequently, the actuators of the system, the motors, required some form of positional feedback. Initially, small remote controlled (RC) servo motors were chosen as they were easily sourced and relatively inexpensive, while still having the ability to be position controlled. One problem was that RC servos traditionally have a rotational range of less than 360° due to mechanical restrictions on the integrated positional feedback potentiometer. This was inadequate for the cable spooling requirements of µAngelo. This problem was solved by the selection of a sail winch servo motor, which is designed to have a
rotational range of several full turns. The selection of sail winch servos was advantageous for two reasons; firstly, these motors allow for a rotational range of up to about ±4 full turns (8 turns in total), which allows for the proper spooling of cables; secondly, these motors come equipped with a winch drum (spool), thus providing a complete packaged solution for a motor and spool.

The specific component chosen for use in the μAngelo system was the Hitec HS-785HB sail winch servo with the specifications as shown in Appendix B.1.

4.1.2 Motor Configuration

μAngelo was chosen to be a three-cable-driven parallel planar manipulator with motors in a Y-configuration as in Figure 4.1. This configuration was chosen to mimic PICARSO (see Section 0, Mechanical System).

![Figure 4.1: An image displaying the three motor Y-configuration.](image)

4.1.3 Motor Controller

The motor controller specified to be used with μAngelo system was a PicoBytes PicoPic, which provided servo motor control for up to twenty servo motors via a RS232 serial connection. The technical specifications of this component are detailed in Appendix B.2 (PicoBytes Inc 2005).

4.1.4 Cables and End-Effector

The specifications for the cables and end-effector were designed to sustain approximately 300g. As a result, the cable chosen for the μAngelo system was Tiger Tail, which was a nylon coated
stainless steel wire typically used by beading hobbyists. The end-effector was constructed out of fencing wire bent into a triangular shape with three mounting holes to attach the cables as shown in Figure 4.2. Additionally, tension springs were attached between the end-effector and the top two cables to protect the motors in the event of a motor command failure or incorrect programming. The springs also acted as a visual indication as to whether the motors were tensioned evenly during operation.

![Figure 4.2: A photograph of the μAngelo end-effector.](image)

### 4.1.5 Mounting Board
The mounting board for the μAngelo system was made out of 12mm plywood and the motors were secured to raised mounting holes in a Y-configuration. Additionally, a foldable triangular rear stand was added to the mounting board to allow the board to stand in a vertical position for operation or to be folded away for storage, thus improving portability of the system. Final hardware design configurations can be seen in the Figure 4.3 and Figure 4.4.

![Figure 4.3: Photographs of the μAngelo motor mounts showing (a) the side view and (b) the front view.](image)
4.2 Software Design

The software design of μAngelo consisted of the control software for the system as well as an integrated image processing program. Since the motor controller chosen was a PicoBytes PicoPic, the control software was responsible for the lower level motor control commands as well as a simple higher level user interface for intuitive control of the end-effector. The PicoPic interfaced to a personal computer via an RS232 serial interface and so MathWorks MATLAB was chosen as the development environment for the control software. RS232 functionality was realised by utilising the in-built RS232 toolbox to send commands to the motor controller, while a MATLAB’s user input function was used to create command prompt type user interface.

4.2.1 Control Software

Two main programs were written for the control of the μAngelo system: a simple point-to-point control of the end-effector and a point-to-point control with trajectory path interpolation between points for smooth operation. The structure of both programs can be seen in the software flow diagrams as in Appendix B.3, while the software code can be found in Appendix B.4.
The structure of both programs began with the initialisation of the serial port and positioning of the end-effector module at the origin of the workspace. The initialisation was then followed by the main while loop function of the code, which included the inverse kinematics calculations and sending of the motor commands for each new user input. Upon a user command to exit, the program would break from the while loop, reset the position of the end-effector to the origin and close the serial port.

The difference between the two control software programs occurred in the main while loop function. Within the point-to-point control program, the main while loops asked the user to input x and y-coordinates for the next position of the end-effector. From these values a new set of motor commands were calculated and sent to the PicoPic. In the point-to-point control with trajectory path interpolation program, the main while loop incremented the current coordinate point to the next coordinate by the predefined increment resolution, recalculated and sent the new motor commands, and then repeated this until the end-effector reached the desired coordinate point. Once this point was reached, the user was prompted to input a new point in the same fashion as the point-to-point control program until a user command to exit. This trajectory path interpolation was a very coarse method of interpolation and was only effective for horizontal paths, vertical paths and 45° angle paths. The reason for this was due to the increment method which utilised a set increment resolution, however if a variable increment resolution was used, that is proportional to the distance between the current x or y-coordinate and the next x or y-coordinate, angle paths other than 45° could have been created. This ability however, was deemed unnecessary for the testing and evaluation objectives associated with the μAngelo system and a 45° angle trajectory path was considered sufficient.

4.2.2 Inverse Kinematics

The inverse kinematics component of the control software was required in order to calculate the required motor position commands for any given coordinate point, or rather end-effector position, within the defined workspace. For this to be done, the workspace, reference frames and relative positioning of the components of the μAngelo system must be defined.
Figure 4.5: A diagram which defines the workspace and reference frame for µAngelo, where *CP is the end-effector centre point and **the theoretical workspace defines all points that are theoretically obtainable for the end-effector centre point.

The inverse kinematics can be derived with reference to Figure 4.5, with the reference frame dimensions assumed to be 670×670mm. These dimensions correlate to the reference frame dimensions on the physical µAngelo system, which are represented in Figure 4.4 (Front View) by the blue dashed square. Furthermore, the value of 670mm was obtained by calculating the size of the square whose diagonal length is the maximum possible cable length that the sail winch servos can spool, given that they have a tested winch travel of ±4 turns (8 turns in total). The calculations for the reference frame dimensions and inverse kinematics are detailed in Appendix B.3.

Parameters used in Figure 4.5 are defined below and were be used to derived the inverse kinematics equations for the µAngelo system.
• $L_1$ – Length of cable 1
• $L_2$ – Length of cable 2
• $L_3$ – Length of cable 3
• $W$ - Half-width of the canvas
• $H$ – Height of the canvas
• $A$ – Upper-cables end-effector centre offset
• $B$ – End-effector half-height
• $x$ – $x$-coordinate of end-effector position
• $y$ – $y$-coordinate of end-effector position

Utilising the parameters above, along with Figure 4.5, the inverse kinematics were derived (see Appendix B.5 for calculation details). The required length of cable 1

$$\Rightarrow L_1 = \left[ x^2 + \left( y - B + \frac{670-H}{2} \right)^2 \right]^\frac{1}{2}, \quad (4.1)$$

while the required length of cable 2 is

$$\Rightarrow L_2 = \left[ \left( W - x - A + \frac{670-2W}{2} \right)^2 + \left( H - y - B + \frac{670-H}{2} \right)^2 \right]^\frac{1}{2}, \quad (4.2)$$

and the required length of cable 3 for a given $x, y$-coordinate, within the workspace, is

$$\Rightarrow L_3 = \left[ \left( W + x - A + \frac{670-2W}{2} \right)^2 + \left( H - y - B + \frac{670-H}{2} \right)^2 \right]^\frac{1}{2}. \quad (4.3)$$

The inverse kinematics equations, above, provided the calculations for the cables lengths required to position the end-effector at any given coordinate point within the workspace. These cables lengths can then be converted to the appropriate motor position values, the equations and explanations of which are detailed in Appendix B.5.

**4.3 Testing and Results**

Testing was undertaken for the μAngelo system with three objectives in mind. The first objective was to analyse and develop a better understanding of the operation of a cable-driven parallel planar manipulator system. The second objective was to confirm the kinematics and analyse the viability of a three motor configuration, while the third objective was to evaluate the system for any limitations and implications which could be carried through to the full scale system. In order to test these three objectives, a specific testing procedure was developed and completed on the μAngelo system.
Three major tests were performed on the μAngelo system utilising the point-to-point control with trajectory path interpolation program. This program was used to allow the end-effector to move around the workspace in horizontal, vertical or 45° angle straight paths, which provided the system with the ability to execute simple trajectory paths for testing. The point-to-point control program was not utilised because the trajectory path was difficult to predict and control. This was a result of a lack of motor speed and acceleration control, and inadequate motor synchronisation. If the point-to-point program was used, inaccuracies such as these may have hindered testing result. A summary of the testing these procedures implemented using the point-to-point control with trajectory interpolation program, are as follows.

4.3.1 Test 1: General Movement
This test aimed to move the end-effector over several points to create a variety of paths across the workspace and to address the following issues:

- Ability to execute simple straight line trajectory paths
- Ability to track along the anticipated path between position points
- Stability along simple trajectory paths
- Evaluation of the kinematics
- Evaluation of executing horizontal, vertical and 45° angle trajectory paths
- Reproducibility of the same trajectory paths

A visual representation of this test can be seen in Figure 4.6.
Test 1 Results Summary

The results for the general movement test are summarised in Table 4.1.

Figure 4.6: Diagrammatic representations of the general movement tests for (a) horizontal paths, (b) vertical paths, (c) forty-five degree diagonal paths and (d) various trajectory paths.
Table 4.1: Summary of testing results for test 1.

<table>
<thead>
<tr>
<th>Test Path</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Horizontal Straight Line</td>
<td>• Accurate and reproducible</td>
</tr>
<tr>
<td></td>
<td>• End-effector tilts inwards as it moves either side of the y-axis (x = 0)</td>
</tr>
<tr>
<td></td>
<td>• Inside cable not as tensioned as expected</td>
</tr>
<tr>
<td></td>
<td>• CP tracks relatively accurately along anticipated path</td>
</tr>
<tr>
<td></td>
<td>• Tilting does not seem to affect the positioning of the end-effector</td>
</tr>
<tr>
<td></td>
<td>• Bottom cable is loose as it moves either side of the y-axis (x = 0), but tightens as end-effector moves up the workspace</td>
</tr>
<tr>
<td></td>
<td>• End-effector vibrates under tension at highest horizontal line</td>
</tr>
<tr>
<td>Vertical Straight Line</td>
<td>• Accurate and reproducible</td>
</tr>
<tr>
<td></td>
<td>• End-effector tilts inwards on either side of the y-axis (x = 0)</td>
</tr>
<tr>
<td></td>
<td>• CP tracks relatively accurately along anticipated path</td>
</tr>
<tr>
<td></td>
<td>• Bottom cable is loose on either side of the y-axis (x = 0), but tightens as end-effector moves up the workspace</td>
</tr>
<tr>
<td>45° Angle Straight Line</td>
<td>• Accurate and reproducible</td>
</tr>
<tr>
<td></td>
<td>• End-effector tilts inwards on either side of the y-axis (x = 0)</td>
</tr>
<tr>
<td></td>
<td>• Bottom cable is loose on either side of the y-axis (x = 0), but tightens as end-effector moves up the workspace</td>
</tr>
<tr>
<td>Various Straight Lines</td>
<td>• Accurate and reproducible</td>
</tr>
<tr>
<td></td>
<td>• End-effector tilts inwards on either side of the y-axis (x = 0)</td>
</tr>
<tr>
<td></td>
<td>• Bottom cable is loose on either side of the y-axis (x = 0), but tightens as end-effector moves up the workspace</td>
</tr>
</tbody>
</table>

4.3.2 Test 2: Accuracy

This test aimed to move the end-effector to specific positions on a grid defined within the workspace and to address the following issues.
• Positioning accuracy relative to the grid points
• Reproducibility of positioning

A visual representation of this test can be seen in Figure 4.7.

![Figure 4.7: A diagrammatic representation of the position grid used for the accuracy test.](image)

Test 2 Results Summary

The results for the accuracy test are summarised Table 4.2.

<table>
<thead>
<tr>
<th>Test Path</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>100mm Movement from Point-To-Point</td>
<td>• Accurate point positioning</td>
</tr>
<tr>
<td></td>
<td>• Reproducible positioning</td>
</tr>
<tr>
<td></td>
<td>• Accurate movements from one point to next horizontally, vertically and diagonally</td>
</tr>
<tr>
<td>50mm Movement from Point-To-Point</td>
<td>• Accurate point positioning</td>
</tr>
<tr>
<td></td>
<td>• Reproducible positioning</td>
</tr>
<tr>
<td></td>
<td>• Accurate movements from one point to next horizontally, vertically and diagonally</td>
</tr>
</tbody>
</table>
4.3.3 Test 3: Workspace Limits

This test aimed to move the end-effector to the theoretical workspace limits and to address the following issues.

- Ability to reduce theoretical workspace limits
- Reproducibility of reaching workspace limits
- Evaluation of the effective workspace for the system

The theoretical workspace limits and effective workspace are defined by the red and green dotted lines respectively as in Figure 4.5.

Test 3 Results Summary

The results for the workspace limits test are summarised in Table 4.3.

<table>
<thead>
<tr>
<th>Test Path</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left and Right Workspace Limits</td>
<td>• Reached close to the theoretical limits</td>
</tr>
<tr>
<td></td>
<td>• End-effector was virtually hanging from one cable, however the other two cables did provide some tension which pulled the module slightly inwards from the limits</td>
</tr>
<tr>
<td>Upper Workspace Limit</td>
<td>• Reached the upper theoretical limits at the left and right workspace extremes</td>
</tr>
<tr>
<td></td>
<td>• Could not reach the centre upper limit, as expected, because an infinite mount of tension force is required to reach these points from a physical point of view</td>
</tr>
<tr>
<td>Lower Workspace Limit</td>
<td>• Reached the lower theoretical limits</td>
</tr>
<tr>
<td></td>
<td>• Bottom cable was completely loose</td>
</tr>
</tbody>
</table>

4.4 Discussion and Evaluation

The testing procedure implemented on the µAngelo system aimed to address three objectives. The first test aimed to move the end-effector over a variety of horizontal, vertical and 45° angle straight paths. The results obtained from the testing indicated that the end-effector tracks along these paths with a high degree of accuracy. A high level of accuracy was maintained over the
same path and different paths. However one issue which was experienced concerning the general movement of the end-effector was that the end-module tended to tilt inwards as it moved on either side of the central y-axis. Furthermore, this tilting increased as the end-effector moved further away from the centre. Additionally, it was found that the bottom cable became loose relative to the other cables as the end-effector moved to the sides of the workspace, but became in tension again as the end-effector moved up the workspace. Despite these issues, positioning of the end-effector seemed unaffected and the centre point tracked the anticipated path relatively accurately. One other issue identified was that the end-effector vibrated quite significantly under high cable tension when the end-effector was positioned close to the top of the workspace. This effect was believed to be created by the end-effector springs and an increased tension load.

The second test aimed to move the end-effector over a series of specific grid positions defined within the workspace. The results indicated that the positioning of the end-effector on 100mm and 50mm spaced grid points was very accurate and consistently reproducible. The end-effector was able to move accurately, without failure, from one grid point to the next as specified by the user. Additionally, the slight tilting of the end-effector on either side of the centreline did not seem to affect the positioning of the module.

The third test aimed to obtain an estimate of the effective workspace of the µAngelo system. The results indicated that the end-effector was able to reach very close to the theoretical workspace limits on both the left and right hand side and bottom edge of the workspace. However, the end-effector was unable to reach the centre workspace limit for the top edge due to the fact that the required forces on the cables, to reach this point, were beyond the limitations of the motor as shown in Figure 4.8.

![Diagram showing the comparison of the effective workspace of µAngelo (green line) against the theoretical workspace (red line).](image-url)
The results obtained from all three tests allowed for the development of a better understanding of the operation of the μAngelo system and in general, cable-driven parallel planar manipulator systems. The results indicated that such systems have the ability to be accurately positioned and can execute simple trajectory paths. The kinematics of the system were verified and effective in the positioning of the end-effector, while the three motor configuration proved to be a physically viable approach for a cable driven manipulator system. The system, however, was not without limitations. One limitation identified was that the effective workspace of the end-effector was slightly less that the theoretical workspace. The end-effector was not able to reach the very top edge of the theoretical workspace, due to limitations on motor torques. However the end-effector was able to reach the bottom edge of theoretical workspace and could move very close to the side edges. The same workspace limitation will be imposed on the full scale system and so the implications of this on the maximum effective drawing area must be taken into account. Further implications on the full scale system include the tilting of the end-effector and the loss of tension in the bottom cable, where the end-effector moves away from the central y-axis, as well as vibrations in the end-effector module that began to occur under high tension of the upper cables.

The degree to which the discussed limitations and issues affect the full scale system is unknown as of yet and may be minimal due to the full scale system utilising much higher quality motors and more complex and effective control software. The accuracy of the trajectory path tracking and positioning of the end-effector of the μAngelo system provided verification of the kinematics and viability of a three motor configuration. Overall the testing of the μAngelo system provided a better understanding of a three-cable-driven parallel planar manipulator system, which shows that the system is feasible to be designed and built at a large scale.
5 Full Scale Design Development - PICARSO

The design of the full scale PICARSO system has been significantly influenced by the outcomes of the µAngelo prototype. The role of the µAngelo prototype was to act as a proof of concept to validate the design choices that have been made. As a result of the µAngelo testing, the tri-motor configuration design will be maintained. The limitations of this construction, as previously discussed, can be minimised with the use of higher quality motors. By retaining the motor configuration, the inverse kinematics of the full scale PICARSO system would be very similar to that of µAngelo, which will maintain the simplicity and ease of knowledge transference between the two systems.

PICARSO’s design process has been divided into the sub-systems that are required to achieve the design specifications and functional requirements to fulfil the project’s overall design objectives. The sub-systems incorporate the main systems of design: mechanical system, including the design and specification of the cables, winding drum, motor mounts, motor assembly and controller; painting system, including the spray gun, tubes, compressor, and solenoid; and the control system, including the image processing and control software. In addition to the functional requirements of the full scale PICARSO system, as a condition to acquiring sponsorship from the University of Adelaide Creativity and Innovation Fund for Open Day, the final design must appear professional and aesthetically pleasing as a promotional tool.

The design development process of these sub-systems, including concept solutions, detailed designs and progress evaluations for each sub-system will be detailed as follows.

5.1 Image Output Type

There were two main decisions concerning the type of image which would be produced by PICARSO. The first decision involved deciding the approach for image representation, in vector or raster method, while the second decision involved whether the colour style of the image will be colour, greyscale or binary. Many considerations were taken for both these decisions, since they were critically important in affecting the final image which was produced by PICARSO.

5.1.1 Raster versus Vector Imaging

Raster imaging is a pixel-by-pixel based approach. Images are represented by grids, where individual pixels are filled in to form the image. Raster images are used in printers and standard picture formats such as JPEG and GIF files. The advantage of a raster approach was its ease of application to PICARSO. A systematic approach could be taken when designing the trajectory
system, which would involve the printing mechanism moving to pixels which required painting. However, the quality of raster images is highly dependent on the image resolution (the number of pixels forming the image). Resolution was a limiting factor for PICARSO, since having a higher resolution would take much longer to process and paint. Furthermore for a higher resolution image, the system would have to be much more accurate to avoid pixel overlap. Even a small overlap would have adverse effects on the total picture quality.

The other picture type considered was a vector image. In this case, the desired picture is defined by lines and curves rather than the individual pixels. This method was deemed as the more elegant method, which could be implemented to mimic a real arm drawing an image. MATLAB includes an image processing toolbox had features which could find the edges in raster form as described in the function reference list by MathWorks in 2010. The major problem associated with this approach is that algorithms would need to be developed to convert edges and other features of the image to a suitable form to be read by the main controller such as a set of equations of the curves. However once this was done, compared to a raster approach, the amount of information which was sent to the main controller would be significantly reduced, since blank pixels of the canvas are ignored. Furthermore the actual painting process would be much faster.

A comparison of a vector and raster image is shown in Figure 5.1. It can be seen that when zoomed in, the image quality is significantly reduced for a raster image. Since the vector image is represented by mathematical equations, it retains its crispness.

![Figure 5.1: Vector (left) and raster (right) comparison (Wiley Printing 2008).](image)

It was decided that PICARSO would initially paint a raster image, as the concept was relatively straight-forward and was a minimal risk approach. Once this program was completed, a vector based solution would be developed. It was thought that this method would require further
research and many different approaches would need to be undertaken in order to find the most efficient solution.

5.1.2 Colour, Greyscale and Binary

The decision of colour style was mainly dependent on the capabilities of the spraying mechanism. The simplest image which was considered was a binary (two colour, commonly black and white) image. This type of image was monochromatic with either a logical 1 to represent a coloured in pixel or 0 otherwise. The next improvement from a binary image was a greyscale image, which was still monochromatic, but took into account the darkness of the pixels or lines.

Finally, the best possible image involved full colour. However, producing a full colour image was deemed to be too complex, since the mixing of colours and different shades would need to be considered, much like a print head. Although a set number of colours could have been used, in this case it was likely that additional spray guns would need to be purchased for each colour to distribute the paint. Therefore due to cost, time constraints and complexity, producing colour image was not considered for PICARSO. Ultimately, it was decided that initially a binary image would be produced and depending on its success, a greyscale image would be attempted as an extension.

5.2 Hardware

This section will discuss the design choices relative to achieving PICARSO’s overall design specifications and functional goals. The hardware design includes the mechanical components and electrical circuitry within the system.

5.2.1 Mechanical System

The mechanical system pertains to the hardware of the system excluding the painting module end-effector and electrical components. The mechanical system incorporates the design of the upper left, upper right and lower central motor wall mounts. This also includes the winding drums and spooling mechanism, cable selection and the motor, gearhead and encoder selection.

Cable Configuration

The end-effector was suspended and manipulated by a number of cables. Due to the tri-motor configuration specification, the minimum number of cables to the motors was three. The number of cables has been determined by the number of motors and also determines degree of control of the pitch and yaw of the end-effector.
The two upper motors allowed for control of displacement in the x-y plane, and the lower motor allowed for acceleration in the y-direction faster than gravity. Furthermore, the roll component of the end-effector was stabilised by the lower motor by manipulation of the cable length.

Drawing from the six DoF platform control of the NIST RoboCrane (Albus et al. 1992), the pitch and yaw of the end-effector was able to be controlled by a second set of cables attached in a different plane. A six DoF system would be a more stable solution and it could be possible for the PICARSO system to achieve this with six individual motors and spooling systems. The stability of six DoF control could also be implemented by grouping the motors in parallel pairs in the Y-configuration. By maintaining the cables to a fixed length by spooling at the same rate, the yaw and pitch will not change, even with an uneven distribution of mass about the centre of gravity. This was also possible with the use of three motors rather than six to retain the simplicity of the system.

**Winding Drum Fabrication**

The winding drum design was dependent on the wire configuration selected. A single drum for the spooling of each cable would be appropriate when using three cables and three motors or using six cables and six motors for 6 DoF control. Additionally, if the design was to utilise two parallel cables, either both cables could be spooled from the same drum or two single spooling drums can be coupled together.

Various options were considered for the material and manufacturing processes of winding drum. A cylinder manufactured from aluminium, brass, or PVC would be tough, light and capable of withstanding the torque induced by the motors. These materials have the advantage of being readily accessible and easily modified without the assistance of the workshop.

**Winding Drum Orientation**

Three possible motor orientations based on the axis of rotation in the x-direction (horizontal), y-direction (vertical) or z-direction (perpendicular to the working plane) were considered. The motor was orientated with the axis of rotation in the z-plane, which allows for the direct winding along the line of the spooling mechanism (see Figure 5.2a). A minimal number of wire guides to direct the cable to the spool would be required, but the motor protrude would from the wall. This would require a plate to be fixed parallel to and beyond the working plane to fasten the motor.
The choice to spool about the x or y-plane was insignificant, as the same number of components that change the cable direction would be required and the motor could be fastened to the spraying plane with a 90° angle bracket (see Figure 5.2b and Figure 5.2c).

![Spool orientation in (a) z-axis rotation. (b) x-axis rotation. (c) y-axis rotation.](image_url)

**Figure 5.2:** Spool orientation in (a) z-axis rotation. (b) x-axis rotation. (c) y-axis rotation.

**Cable Reorientation**

Literature demonstrated that many cable-driven manipulators employed pulleys to reorientate the cable direction for ease of spooling and convenient motor placement. Pulleys would be highly advantageous due to their simple implementation, but many would be required if the spool was to rotate in the x or y-plane. A second consideration was the use of eyelets, such as eyebolts, to allow for multiple plane orientation of the lines. Eyelets were limited by the diameter and coating of the line, and could result in the line catching if used with an inappropriate cable.

**Winding Drum Operation and Mounting**

The winding drum could be driven by implementing either direct drive or indirect drive. If operated by direct drive, the drum would require a hole for the motor axle and keyway. Supports would require construction to prevent undesirable bending moments on the motor gearhead. This design required either a bearing fastened in the spool with a fixed protruding axle, or a bearing fixed to the enclosure with a rotating axle attached to the drum. Conversely, the indirect drive design considered two bearings supporting an axle separated from the motor, using a chain, belt or gear system to drive it. Although indirect drive design would use more components and increase difficulty and expense, it would allow the spool to be orientated in any fashion desired. Thus, implementing indirect drive allows for the spool to be placed in an orientation that removes the need for changing the direction of the cables, which in turn allows the cables to spool directly.
Enclosure and Wall Mounts

To meet the objective of transportability, each motor, spool, motor controller and cable guide assembly needed to be housed within an enclosure. One option considered was to mount all necessary components within a polycarbonate (PC) enclosure with transparent lid. This solution would allow easy portability of the system and the transparent lid would allow viewing of the components during operation, which would be advantageous during demonstrations. Prefabricated enclosures would be purchased that would require modifications including: holes to mount cable guiding equipment (pulleys, eyelets, etc.), holes for the supporting cables, exits for electronic wires and power supplies, and facilities to install wall mounts. An enclosure allowed for any orientation of the motors but may fracture under the mass and pull of the motors. Furthermore, the enclosure would need to be of sufficient size to house all required equipment.

Further consideration was given to the surface the motors would be mounted on. A metal plate was considered as it would provide an adequate mounting solution for the motors and being thick enough to be less likely to fracture than PC. The plate could also be orientated in any plane to achieve the desired spool orientation and would be mounted within an enclosure that was custom built or purchased, and would require less modification than a prefabricated box.

For normal operation, the design objectives required the PICARSO system to be mounted onto the working vertical surface. It was assumed that wall modifications would be permissible, before PICARSO is first operated. Thus, it was considered that the mounts would attach to bolts installed to the working plane. It was also assumed that during the testing of the system, wall modifications to University of Adelaide wall surfaces would not be permitted, and so a solid timber frame would be constructed to allow for simulated attachment to the wall.

Cable Type Options

The cable selection depended on the mass of the painting module, diameter of the winding drum and cable configuration. The considered cable types included fishing line, braided rope, and braided stainless steel.

The mass of the spray gun and housing was specified to be at most 3.0kg without paint flow. Considering the extra weight of to the paint during operation, and applying a safety factor of 3.0, the maximum mass suspended by a single cable in a worst case scenario, where only one cable supports the painting module, was determined to be 10kg (98.1N). Therefore if two cables in
parallel were used, then the effective mass suspended by a single cable at this point was 5kg. Additionally, the chosen cable was required to be flexible in order to wind around the spool and cable reorientation system without knotting, tangling, or causing considerable wear to the components.

Fishing line was considered as an appropriate cable choice for the PICARSO system primarily due to its availability in different strengths, its light weight and ability to slide through eyelets with minimal friction. Standard nylon fishing line was deemed to be too elastic, but Dacron (Polyethylene terephthalate) and braid (including Spiderwire) were less elastic and possessed a higher breaking strain. However, the Dacron and braid lines were also observed to be very fine and hence had the potential to wear down the pulleys and eyelets.

Braided rope was deemed to be advantageous over laid or twisted rope due to its greater ability to uncoil and lack of inherent twist. Braided rope has a greater diameter than the fishing line, and so it would be easier to handle during the assembly of PICARSO. However, braided rope would require a feeding mechanism onto the spool to prevent the cable overlapping, which would increase the diameter of the spool and change the kinematics to a dynamic function. It was also noted that braided ropes were generally made from nylon or polyester which would thread through eyelets with minimal friction, however, its elasticity was undesirable.

The µAngelo system used nylon coated braided stainless steel cable which was effective in suspending the end-effector. The mass of PICARSO’s painting module was designed to be light enough to be supported by this type of cable, however, the mass of the cable would affect the dynamics of the system over a 5.0×5.0m area. From experience with µAngelo, the wire was prone to kink after multiple winds, which would be undesirable for PICARSO.

**Mechanical System Detailed Design**

The end-effector was designed to be suspended by three parallel pairs of cables, arranged in a Y-configuration. The cables will be located at the back and as near as practical to the front of the end-effector, to eliminate undesirable pitching and yawing during operation. It was decided the motors and associated components would be bolted to 6mm aluminium base plate. The base plate would allow for multiple orientations, if necessary, and could be enclosed by a Perspex cover. The aluminium base plate would be adequate to support the mass of the motor without buckling, and is easily workshopped.
The winding drum was selected to be 45mm diameter solid aluminium bar with two 10mm deep channels to separate the cables. The aluminium was deemed the most appropriate material in comparison to PVC and brass in terms of weight, strength, cost, ease of fabrication and aesthetics. The maximum drum diameter of 45mm was selected as it is also the maximum diameter of the motor and allowed for clearance should the motor placement need to be changed. Direct drive was selected to minimise the parts required to support the drum, but required the drum to be manufactured with a 12mm diameter hole for the motor axle and a 4mm keyway. A bearing support will be utilised to reduce the axial load on the motor and will be mounted to the aluminium base plate. The motor would be bolted to the plate with a 6×50×25mm 90° aluminium angle bracket.

The spool will be oriented vertically to rotate about the y-axis on the upper motor mounts and horizontally to rotate about the x-axis on the lower motor mount. Rotation about the z-axis, requiring the motor to extend from the wall, was deemed too complicated to support and not aesthetically pleasing due to the length of the motors. The design required pulleys for the upper left and right mounts, and fishing line was chosen to allow two eyebolts to direct the cable onto the winding drums. The centre mount would guide the line with two eyebolts per cable to the winding drum. To economise parts, the design allowed for the motors, pulleys and eyebolts to attach to the same angle bracket.

Finally, the plate would require holes so the hardware could mount to the walls with bolts. The plate would sit flush on the working surface, with spacers employed to reduce any surface irregularities. Also, the timber frame designed for testing matches these specifications. To minimise visual clutter by concealing the electrical wires, the EPOS2 board has been designed to be elevated approximately 70mm above the plate with computer spacers.

The Computer Aided Design (CAD) drawings of the left, right and centre motor mounts can be seen in Figure 5.3, Figure 5.4 and Figure 5.5 respectively. The holes to mount the base plate to the working surface are visible in Figure 5.3b.
Figure 5.3: Left motor mount CAD design. (a) Isometric front. (b) Isometric back. (c) Labeled front wireframe view.
5.2.2 Motor Assembly

The motor assembly incorporates the motor, gearhead and encoder. This section will discuss the motor type selections and the eventual design choices behind the selection of these components relative to the design specifications and functional requirements of the system. Additionally, the alternative concept solutions will be discussed and evaluated for each of the motor assembly components.

Motor Type Selection

The motor drive solutions implemented by the aforementioned existing painting robots and cable driven manipulators, as discussed in detailed the Literature Review (as in Chapter 2), varied
greatly depending on the requirements of the system. The Portrayer’s motor torque requirements were low and the feedback control was not needed and therefore Lego motors were sufficient for this application. Additionally, the level of control and accuracy provided by Lego’s controller and software package was adequate for this design. In comparison, due to Viktor’s high design requirements, servo motors were utilised. Viktor was the successor of the stepper motor driven Hektor system. Thus, as the project developed in complexity, the requirements of the equipment increased. The implementation of servo motors was needed for smoother motion and feedback control. This feedback control allowed both the system and the user to have localisation awareness. PICARSO aimed to have much greater accuracy and the motor requirements were much greater than in the Portrayer. Therefore, Viktor’s motor system was more applicable to this project.

From the evaluation of existing painting robots and cable driven manipulators, the motor type for the PICARSO system was selected to be servo motors. The requirements and the objectives of the PICARSO system are higher than many of the existing systems and thus, PICARSO’s hardware choices were comparatively more complex. The use of servo motors in the PICARSO system allowed for smoother and more stable drive of cables and a higher level of control, through the use of intelligent motor controllers. By using intelligent motor controllers, the project was able to incorporate the specified feedback control components. Consequently, intelligent motor controllers were essential.

Both Maxon and Aerotech were considered as possible motor and motor controller suppliers. Thus products from both manufacturers were researched to identify the most suitable solutions for PICARSO. Ultimately, the PICARSO project was able to secure sponsorship from Maxon Motors in the way of a discount on products and the provision of support in motor and controller selection. Subsequently Maxon products were chosen exclusively for this project.

When choosing the specific type of motor, the main consideration was whether the torque requirements of the system would be met. The operational speed, which is often a deciding factor in motor selection, was not as important in PICARSO’s case. From the design specification, the minimum torque requirement is 3Nm. Also, the design operation speed was chosen to be 150rpm at the spool. However, in the final design, if a concession needs to be made between the speed, torque and accuracy of the spray gun output, the speed of the motors can be compromised. Motors were then selected to achieve these specified requirements.
Motor Torque Requirement Calculations

The main consideration for the choice of motor size in the PICARSO system was torque, which indicates the rotary force that is produced on the motor shaft. In the worst case scenario, a single motor in the PICARSO system would be required to support the entire weight of the painting module. The estimation for the total weight of the painting module, with additional safety factor included, is 3kg. Taking acceleration due to gravity to be 9.81ms\(^2\) and the radius of the spool to be 0.1m, the maximum required torque will be 3Nm. When selecting the motor, an additional safety factor was added to the calculated required torque of up to 5Nm, to provide flexibility with the design of the painting module. If additional components in the future design processes are required to be housed on the end-effector, the motors would be sufficiently sizes to accommodate for this.

Maxon’s EC series of servo motors are electronically commutated brushless DC motors. This series of motors was selected because their operation ranges lie within those described in PICARSO’s project specification. Moreover, Maxon’s EC motor range provided numerous gear, feedback components and control electronics combination choices. To allow for the motor to take the specified 3Nm or maximum torque of 5Nm, the motor power would need to be at least 150W. The motors from Maxon's range that could meet these requirements and that were also verified by Maxon support staff were the EC45 150W and the EC45 250W.

The performance curves for the Maxon EC45 150W and 250W motors are shown in Figure 5.6 and Figure 5.7.

![Figure 5.6: Maxon EC45 150W motor performance curve which showing the torque rating. (Maxon Motors Australia 2010).](image-url)
The performance curves represent the potential outputs of each type of motor. As motors are designed to work at a percentage of their maximum rated load, the actual operation properties must be considered when making their selection. If the motors are overloaded in operation, they can overheat, become damaged and lose efficiency, and thus, must be specified properly. The shaded red area in Figure 5.6 and Figure 5.7 indicate the motor’s output capabilities while running in continuous motion. The torque value at the edge of the shaded red area indicates the maximum continuous motion torque. The maximum continuous torque value for the EC45 150W motor is 5.5Nm, operating at approximately 130rpm. Conversely, for the same value for the EC45 250W motor is 6.6Nm, operating at approximately 175rpm.

Ultimately, the motor chosen was the EC45 250W because of the flexibility that a larger motor can provide. The specification of the EC45 250W are shown in Appendix D1. The larger motor meant that there is greater weight allowance for the painting module, which in the heaviest case scenario consists of the loaded painted gun and tubing, actuation mechanism and solenoid.

**Alternative Motor Solutions**

The tri-motor configuration implies that the two motors are located in the upper left and right corners of the canvas and carry the entire weight of the painting module. Ideally, the third motor will be positioned at the bottom of the ‘Y’ and would be identical in specification to the upper two motors for ease of control and consistency in the system. Realistically, the torque requirements of this third motor would be significantly lower as it will not be supporting the
painting module and only provides sufficient torque to oppose the upper two motors and maintain the stability of the painting system.

An alternative motor selection solution would be to reduce the size requirements of the lower third motor. However, under the suggestion of Maxon support, the relatively small financial saving from stepping down the motor would be detrimental to the elegance and robustness of a solution as compared to three identical motors. Additionally, the smaller motors require an enclosure to protect the motors while the EC45 series motor included features which could protect it from damage due to dust, dirt, or paint. Thus, for overall robustness of the design solution, three identical motors in the tri-motor configuration have been chosen.

**Gearhead and Encoder Selection**

The selection of the EC45 250W motor needed to be coupled with the appropriate gearhead and encoder. The motor drive system was chosen based on the previously stated design speed of 150rpm. To achieve an operating speed of 150rpm at the spool, the gear ratio would have to be approximately 30:1. Maxon motor provided compatible gearhead assemblies that were available at 26:1 and 43:1 ratios to achieve this desired speed. Due to the existence of fewer gear stages and the desire to maintain efficiency the gearhead choice was 26:1. Furthermore this would lead to shorter delivery times by selecting a stock part. The specifications for gearhead are shown in Appendix D2. Additionally, Maxon HEDL 9140 optical 500 counts per turn encoders were chosen for compatibility with the motor controllers. The specifications for encoder are shown in Appendix D3.

**5.2.3 Electrical and Electronic System**

The electrical design of PICARSO included the circuitry and wiring required for power distribution to the motors, solenoid and signal routing.

**Power Distribution**

When considering the power distribution of the PICARSO system, the required maximum power capabilities during operation as well as the required voltage must account for voltage drop must be accounted for.

A power supply was required to provide the three EPOS2 70/10 controllers which each supply a 250W $24\text{V}_{\text{DC}}$ motor and the $24\text{V}_{\text{DC}}$ solenoid valve for the compressed air system. Thus, the peak rated power supply was calculated by the summation of the load drawn from these components as given in Equation (5.1), and was determined to be 800W. The 1600W TKD Lambda
programmable power supply unit belonging to the University of Adelaide School of Mechanical Engineering was considered an adequate unit to meet this power requirement.

\[
P_{sys} = f_{safety} \left( 3 \times P_{EPOS} + 3 \times P_{Motor} + P_{Solenoid} \right) \tag{5.1}
\]

Where:
- \( P_{EPOS} \) – Power drawn per EPOS2 controller \( \sim 3 \) W
- \( P_{Motor} \) – Power drawn per motor = 250 W
- \( P_{Solenoid} \) – Power drawn by the solenoid \( \sim 1.5 \) W
- \( f_{safety} \) – Safety factor = 0.05

To accommodate for voltage drop during the operation of the motors, the supply voltage to the EPOS2 70/10 controllers was required to be greater than 24V. This was calculated by

\[
V_{CC} = \frac{U_N}{n_0} \cdot \left( n_B + \frac{\Delta n}{\Delta M} \cdot M_B \right) \cdot \frac{1}{0.9} \cdot 1[V] \tag{5.2}
\]

as given by the EPOS2 70/10 data sheet (Maxon Motors Australia, 2010); where
- \( M_B \) – Operating torque = 310mNm;
- \( n_B \) – Operating speed = 4520rpm;
- \( U_N \) – Nominal motor voltage = 24V;
- \( n_0 \) – Motor no-load speed at \( U_N \), = 5250rpm;
- \( \Delta n/\Delta M \) – Speed/torque gradient of the motor = 2.19rpm/mNm.

Applying a safety factor of 0.2, the required supply voltage was determined to be 33V_{DC}.

**Signal Routing**

The digital signal connections involved the design of circuitry which connects the EPOS2 controllers with the motor units, communications network and emergency stop capabilities. The EPOS2 70/10 controllers each interfaced with their motors via a +5V_{DC} Hall Effect connection and encoder connection. The lower unit also controlled the solenoid using one +24V_{DC} 500mA digital I/O connection. A switching circuit should not be required for this connection due to the optical isolation provided by the EPOS2 controller on all I/O ports. However, an optical
switching circuit will be implemented should the current required by the solenoid exceed the capabilities of the EPOS2 controller.

The master computer was designed to connect to the RS232 port of the Motor 1 (lower central) controller. Alternatively, an USB connection may be used if the rate of data transfer required exceeds the RS232 capabilities. Motor 2 (upper right) and Motor 3 (upper left) respectively were planned to connect in parallel to Motor 1 via a cable area network (CAN) link for simultaneous operation.

In case of emergency, a switch was considered to completely halt the motor operation of PICARSO. This switch would be normally closed (NC) and located upstream of the 33V\textsubscript{DC} bus which would cut all power to the motors, EPOS2 70/10 boards and solenoid when activated. It was noted that this device would not halt the pressurised air used in the system.

It has been considered for aesthetic design that the circuitry is concealed behind the wooden support frame. For implementation of the PICARSO system without the frame (i.e. directly to the working surface), the wires and data lines would hang between the motors beyond the reach of the cables and end-effector to prevent tangling.

Electrical Circuit Layout
The electrical layout of power and signal connections can be seen in Figure 5.8.
5.2.4 Hardware Design Progress Evaluation

The design and specification of the hardware components has been evaluated as: the progress of the mechanical component design, the motor specification including the motors, encoders and motor controllers; and the specification of the electrical system.

Mechanical Components Design Progress

The CAD design of all of the mechanical components has been completed and the drawings have been ordered through the workshop and are awaiting manufacture. These can be seen in

Figure 5.8: Single line schematic of power and signal connections.
Appendix C. Several revisions of the design of the mechanical system components, specifically the cable reorientation and wall mounting system did take longer than expected, and inexperience in the CAD software package Pro-Engineer Wildfire 5.0 (ProE) eventuated in running over the desired allocation of time defined in the Gantt chart. However, these delays are not expected to impact on the tasks running concurrently with the hardware design and manufacturing, nor on the future tasks depending on them as the components are not required until the beginning of July 2010.

Motor Specification Progress

The motor's functional and design specifications have been identified. Based on the design requirements, the motor, gearhead and encoder assembly have been specified and the order has been placed with Maxon Motors Australia, although the components themselves are assembled and shipped from Switzerland. The delivery dates provided by Maxon state the components will start to arrive from the 13th of May, with the delivery of the cables, until the 24th of June, when the EPOS2 motor controllers will arrive. The anticipated arrival date of the entire motor assembly and motor controller is significantly later than 14th of May, which was the date specified in the Gantt chart for the completion of the installation and connection of the hardware to a computer for testing but this has been unavoidable due to unexpectedly long lead times. Additionally, the delivery date is very close to the University of Adelaide Open Day on August 15th, where PICARSO will need to be fully functioning.

Electrical System Specification Progress

The maximum power required during system operation has been specified from information sourced from the Maxon Motors Australia data sheets and the solenoid requirements. The TDK-Lambda power supply has been sourced from the School of Mechanical Engineering. Furthermore, the required operating voltage has been determined and contingency plans have been drafted should digital isolation circuits be required to drive the motors or solenoid. These tasks were achieved within the desired timeframe.

5.3 Painting System

The purpose of the painting system is to mechanically distribute paint to produce the input image onto a canvas. In relation to the overall system, it is the component that ensures that the output from the image processing is physically produced onto the vertical canvas.
5.3.1 Painting System Concept Solutions

The concept solutions that have been considered for the painting system design were derived from characteristics of Hektor, Robopainter, and further research and brainstorming performed by the group. The following section will outline concept solutions, which led to the ideas for the optimal design.

Painting Mechanism

When considering suitable painting mechanisms, three different types of spraying mechanisms were researched; spray cans, airbrushes and spray guns.  

Firstly, using a spray can as the painting mechanism was considered because of their availability, and simple design and function. Research into this area was initiated through research into existing systems such as Hektor. In a spray can a gas propellant expands when the valve is actuated, and the subsequent pressure drives out the product mixture (such as paint) through the dip tube and out the nozzle as seen in Figure 5.9. A pea in the spray can is used to mix the product mixture and propellant before use.

![Figure 5.9: Cross-section of a spray can showing the functional design (Knulclunk 2007).](image)

However, the use of the spray can as a suitable painting mechanism for the design lacks the control required to produce accurate and precise images on a 3.0×3.0m vertical canvas. Franke and Lehni found with Hektor in 2006 that a spray cannot produce consistently even lines, and may produce bleeding due to excess expelled paint. Moreover, the weight of the spray can decrease as the paint is expelled creating an additional variable that may affect the dynamics of the system. The range of colours and low cost of using spray can is an advantage, however the paint can only be controlled by the single compression of a valve, and the paint distribution relies
purely on the expansion of the propellant. Therefore, a more elegant and efficient system is preferable for the system.

Further research in the area of paint distribution identified airbrushes such as the ones in Appendix E.2 as possible solutions. Airbrushes are small air-operated tools that work by passing a stream of compressed air through a Venturi effect to create a local reduction in pressure. This then allows paint to be pulled from an interconnected reservoir at normal pressure, atomises it, and disperses it with high velocity through a nozzle to the painting area. Most airbrushes do not have adjustable nozzles (to adjust spray pattern), but rather rely on the changing the distance from the canvas to produce different artistic patterns.

Airbrushes have different designs and actuation methods, namely single or dual action. Single action airbrushes have a single trigger which releases air and paint into the airbrush body simultaneously. In a dual action airbrush, air is released by compressing the trigger and paint is applied by sliding the trigger back. This is so a user has more control and can produce different artistic effects. A single action airbrush was considered in the conceptual design for our project because it was suitable for the functional and design specifications, and is simplistic in that it would only require one actuator for operation. Furthermore, a double action airbrush was also considered because it may provide control over the intensity of colour and design patterns for the production of images.

There are several methods to feed the paint into the airbrush, namely gravity fed, siphon fed, and side fed. Gravity-fed airbrushes feed paint from above the airbrush (as seen in Appendix E.2 - Figure E.1), subsequently allowing the use of gravity and less air pressure for operation. Gravity-fed airbrushes are considered superior to other paint feeding mechanisms, because of their ability to produce high quality images through the use of gravity. Siphon and side-fed airbrushes (Appendix E.2 - Figure E.2 & Figure E.3) operate by suction of paint, from a reservoir located below and on the side of the spray gun, using high pressure. This method is used to allow for artists to have a clear view of their working canvas. However, since our design is automated, this feeding method, especially with the requirement of more pressure than gravity fed, is undesirable.

Overall, airbrushes have a significant advantage over spray cans with more control of projecting paint efficiently on a painting area. However, airbrushes are generally used for fine, retouch artwork, rather than large canvases. The majority of airbrushes do not have large paint containers, which usually range from 1/16 oz (1.85 mL) to 4 oz (118.3mL) (Airhead Airbrush
Product Reviews 2010). This is an insufficient amount of paint to achieve a single uninterrupted painting session. Consequently, for larger painting areas and higher volumes of paint an adaptation of an airbrush was identified, namely, a spray gun.

Spray guns are larger airbrushes that produce larger spray patterns over larger surfaces. Spray guns can be either hand-held or automated and have similar paint feeding mechanisms as airbrushes, as well as a pressure fed option. From analysis it was found that a pressure fed automated spray gun would be the most suitable option for the PICARSO design.

Pressure fed spray gun as seen in Figure 5.10 & Figure 5.11 work by feeding compressed air and paint through tubing to a pressurised paint canister. This allows the paint to be located away from the airbrush and thus the painting module retains constant weight while operating. This is a significant design advantage over a gravity fed system, where the painting module would become lighter as it operates and may cause changes to the dynamics of the system.

![Image of a spray gun](image)

**Figure 5.10:** A photograph of a pressure-fed spray gun system which could be applied to PICARSO (China Suppliers 2010).

![Schematic of a spray gun](image)

**Figure 5.11:** A schematic of a typical pressure-fed spray gun showing main components (Way Builder 2010).

When choosing between a hand-held spray or an automated gun, the method of paint actuation was considered. If a hand-held spray gun was purchased, it would require physical actuators designed into the housing to pull back the trigger. This was possible, but would not guarantee the instantaneous actuation of the spray gun. An automated spray gun system as seen in Figure 5.8, however, can be triggered instantaneously by an electronically actuated solenoid valve. Automated pressure fed spray guns are normally used with automated robotic painters and can
be programmed to trigger at high frequencies and speeds. Also, automated pressure fed spray
guns are relatively compact because of their external actuation, which simplifies the housing
design for the painting module. However, similar to hand-held spray guns, automated spray guns
have air and fluid lines connected which may obstruct or hinder movement. Regardless, the
benefits of an automated pressure fed spray gun outweigh the disadvantages, hence making it the
ideal painting mechanism for the painting system of PICARSO.

**Housing**

When considering the ideal housing for the painting module a few basic concepts were
developed. However, in the concept solution stage, much of the detailed design of the painting
module were developed in relation to the spray gun selected (section 5.3.2). Several general
specifications for the housing were made.

Since an automated pressure-fed spray gun was used the housing would only have to enclose the
spray gun itself and does not obstruct the air and fluid lines. A light weight and strong material
was considered for the housing considered. Moreover, the material should be relatively inflexible
and be able to take sufficient compressive and tensile stresses and strains as well as shear stresses
resulting from the connection of wires to the motors. Ideally, the material should be low cost,
and easy to manufacture to our specifications. From these considerations, the material choice for
the painting module is aluminium.

For the shape and orientation of the painting module, a few different designs were considered
and tested in the prototype small-scale design of our system. A mouldable piece of wire was bent
into different shapes to test different concept solutions. From this it was discovered that there
was no difference in using a triangular shape (as in Figure 4.2) or rectangular/square shape
design for the painting module, as the three motor configurations are such that these designs

![Figure 5.12: Anest Iwata automated pressure-fed spray gun system configuration.](image-url)
behaved similarly. Other considerations include making sure the housing does not tilt or have imbalances caused by the connecting wires being not located along the centre or mass.

The aforementioned concept solutions must be considered for the detail design of the overall painting system outlined in the following section.

5.3.2 Painting System Detailed Design

Before deciding on the detailed concept design of the painting system, a suitable pressure fed spray gun must be selected and purchased. From research, there were several large companies that sold high quality and suitable spray guns – for instance, Anest Iwata, Grex, and Richpen. All of these companies are well known for their paint distributing mechanisms (including airbrushes and spray guns), and were researched throughout the selection process. Anest Iwata is considered the best performer and highest quality out of the three companies (Airhead Airbrush Product Reviews 2010), however, it is the most expensive of all brands of spray guns. Alternative spray guns considered for our project throughout the research process were – Grex’s Genisis XG, Genisis XS, and Richpen’s Apollo 113C, Phoenix 213C, Gemini 313 C, Spectra 033G, GP ES-6, Rich-8 RS 506N, 507N, 508N. More details of these air brushes and spray guns can be found in Appendix E.1.

Considering all the previous options, it was found that none were suited to be automated and pressure fed. While searching for suitable spray guns it was found that despite their higher cost, Anest Iwata had a range of suitable automated spray guns that were built to be pressure fed and automated. From these qualities, the cheapest and most compact of the range was considered the most suitable for our project, namely SGA-101 (see Figure 5.13).

![Figure 5.13: A diagram of an Anest Iwata SGA-101 automated pressure fed spray gun which was chosen as the suitable option for PICARSO (Anest Iwata 2010).](image)

When purchasing the SGA-101 automated spray gun, the additional components ordered were the pressurized paint canister, 5 metres of twin-tubing, and fittings. A three-way solenoid valve was donated by APEX Robotics and Automation as a switch for air inlet of the spray gun. The system as whole (Figure 5.12) separates the spray gun from the paint pump and other
components via tubing. This means that the housing for the painting module only needs to account for the spray gun and connecting tubes.

There were three detailed designs considered for the painting module. The following section will outline their design specifications. However, first the detailed design of the spray gun must be identified. Appendix E.3 shows the parts list of the SGA-101 pressure fed spray. To aid in the detailed design of the painting module, a CAD model was designed for the assembled spray gun in ProE (Figure 5.14) using a Vernier Calliper in the housing detailed designs.

![Figure 5.14: A diagram of an Anest Iwata SGA-101 CAD spray gun in isometric view showing its main features.](image)

**Painting Module Design 1**

In the first design a backing frame was designed for the spray gun with three protruding rods in front of the spray gun in a triangular orientation as in Figure 5.15. Moreover, the spray gun was positioned in an x-orientation, where the connecting tubes for the spray gun were located ±45° from the negative y-axis (Figure 5.16). This was done to position the spray gun to minimise obstruction from the fluid and air tube lines. In addition, the x-orientation also minimises rotation of the painting module due to any uneven weight distribution of the tube lines.
Figure 5.16: Paint module CAD design 1 (Front) with old spray gun CAD in X-position (see Appendix E.4: Figure E.6 for preliminary CAD design of spray gun).

The housing allowed for the spray gun to sit in an ‘z’-orientation (out of the working plane) against the front side of the frame, with the three protruding rods, one above the spray gun and two located below the tube lines. These protrusions were made to allow the painting module to rest against the canvas and minimise moments in the z- and y-axes’ relative to the end-effector - defining the x-axis as the tip of spray gun nozzle (end-effector) pointing perpendicular to the canvas surface. However, for this to be done the painting module would require a force from the connecting wires to pull it against the vertical canvas.

Another problem faced from this design concerned the protrusions, which may drag over the previously painted parts of the vertical canvas. This may cause inconsistencies or smudges to the image. To solve this problem, the spray gun was mounted on two connecting brackets that were fixed to the fitting knob of the spray gun. This allowed the spray gun to be translated along the z-axis (i.e. moved closer to the working plane), and hence could be placed in front of the protrusions to stop their contact onto the vertical canvas. These protrusions were retained in this design, but included adjustable brackets to remove them if necessary. Another problem
identified in this design concerned the spray gun, which only connected to the fitting knob. Since the spray gun is only connected at one point, the stresses, shear, and moments from the system may cause difficulties in maintaining its position and orientation in the painting module.

In order to attach the connecting wires to the painting module, a hole was drilled into the back of the frame to fit brackets. These brackets would allow the connection of the wires along the central axis of the spray gun. Moreover, counterweights could be attached behind the brackets on the connecting rod at the back of the frame to orientate the end-effector of the spray gun level with the vertical canvas. Overall, this conceptual design incorporated several desirable design traits, but there were many improvements which could be made. Subsequently, a second design was developed.

**Painting Module Design 2**

The second design was an attempt at a more elegant design to the first. The idea behind it was to place a circular sleeve over the spray gun and drill three holes into that sleeve with a $120^\circ$ separation to attach the three connecting wires to the servomotors (Figure 5.17). It was identified that the fitting knob could be removed, thus allowing for a different method of securing the spray gun to a housing.

![Figure 5.17: Paint module CAD design 2, version 1 with spray gun attached. (a) Isometric front. (b) Isometric back.](image)

Consequently, a sleeve for the spray gun was designed such that connecting wires could be attached close to the centre of mass of the painting module so that there was minimal moment created along the $y$-axis relative to the end-effector. If the paint module dipped a positive angle along the $y$-axis (anti-clockwise) due to the weight of the spray gun, a hole could be drilled into the back of the sleeve to attach a rod and counter-weights to solve this problem. However, this
design still had an issue discussed previously, where errors in the translation of the painting module would occur at the corners of the painting surface due to the connecting wires not being attached to the central axis of rotation of the spray gun.

To solve this problem a second version of the design was developed. In this case, the same sleeve was used but a rod was protruded through the back side, with two sets of circular discs with several holes connected (as in Figure 5.18). This design was less elegant than the first version of design two, but initiated further considerations into the design of the painting module.

Two sets of the circular discs are connected on the rod protruding through the back of the spray gun sleeve, connected with three joining threaded rods (120° apart). These are fixed by nuts and washers to stop their translation toward or away from each other. This component as a whole, of the two circular discs and three joining threaded rods, is able to rotate freely around the rod. The connecting wires are then attached to the remaining three holes (120° apart) on the outer rim of the circular disc. The additional three holes (120° apart) in the inner radius of the circular disc provided additional options for attachment of connecting wires, or further support for the structure. The translation problem is eliminated by the ease of movement of the component around centre of rotation of the spray gun. Also, because the connecting wires are attached to both of the circular discs, this provides additional support that will eliminate moments about the z-axis relative to the end-effector. However from an aesthetics point of view, this design is not as elegant and compact as its previous version. Also, there will be rotational shear problems faced from the circular discs tendency to spin in different directions. These problems prompted a third revision of the design.

Figure 5.18: Painting module CAD design 2, concept 2 with spray gun attached. (a) Isometric Front. (b) Isometric Back.
Painting Module Design 3

This design combined the ideas that were developed from the first and second design. Subsequently, a sleeve similar to design two was proposed, which would fit over the spray gun and extended as in Figure 5.19. Instead of the lotus-disc shaped discs as in second design, holes were drilled directly into the extended part of the sleeve. An initial three holes (120° apart) were drilled at the front end of the sleeve just behind the spray gun. Subsequently, a further six sets of three holes (120° apart and 10 millimetres between each other) were drilled 10 centimetres from the front end holes. The first set of connecting wires will attach to the first three holes, and the second set of connecting wires can be attached to any of the six sets of holes on the back end of the sleeve. This allows for modularity in the length between the connecting wires which could be changed to control the moments in the z-axis relative to the end-effector. It is noted that this design will have translational problems due to the connecting wires’ attachment to the outside of the centre of rotation, however this is a problem which will assessed during testing stages.

This simpler and more elegant third design will be used for our full-scale system, whilst a fourth design will be designed in the future to eliminate this problem.

![Figure 5.19: Painting module CAD design 3 with spray gun attached. (a) Isometric front. (b) Isometric back.](image)

5.3.3 Painting System Progress Evaluation

The paint system progress is on-schedule with regards to the Gantt chart (Appendix A). There were initial setbacks in relation to spray gun selection and their relevant components. But with the help of industry sponsors, the spray gun, connecting tubes and fittings, pressurised paint canister, black paint, and three-way solenoid valve were chosen to the required design specifications. The
full drawings has been developed on ProE for Design 3 and relevant design drawings have been approved and are currently being manufactured.

5.4 Image Processing Design Development

The image processing program was developed in MathWork’s MATLAB in an iterative process, with parts being added according to the design specifications of the program. Once the main functions were added, several theoretical simulations were written in order to observe how the program would be applied to the full scale PICARSO. The details of the program development are described in the following section.

5.4.1 Program Selection - MATLAB

MathWork’s MATLAB program was deemed an appropriate choice as the platform to develop the image processing software due to the availability of the program within the University. This meant that no additional funds would be required to purchase other software. Furthermore, all of the members of the project group had previous experience in using MATLAB. There was sufficient evidence that MATLAB included the necessary capabilities in its toolboxes to achieve all the relevant goals concerning image processing.

The most relevant toolbox in relation to image processing was the “MATLAB Image Processing Toolbox”, of which capabilities are listed on the MathWork’s website. The website states that the “Image Processing Toolbox provides a comprehensive set of reference-standard algorithms and graphical tools for image processing, analysis, visualization, and algorithm development.” (MathWorks 2010). Other features of note included the toolbox’s capabilities to detect and measure features and edge detection. This could especially be applied to the vector imaging extension goal, where it was desirable to isolate outlines of pictures to be painted as a series of curves.

MATLAB is commonly used in robotic visioning applications as shown by Tedder and Jin-Chung in 2004. In this report Tedder and Jin Chung claimed to use MATLAB toolboxes to design a robot visioning system using the image processing toolbox and fuzzy logic toolbox. It was said that “JavaDV is a video capture driver that reads image data from the DV camcorder’s IEEE 1394 Firewire port and returns an array of RGB image pixel values.”(Tedder & Jin-Chung 2004). This system gave evidence that the PICARSO system could utilise MATLAB to take image files and convert them to a matrix or coordinate form. A similar method could be used to interface between the proposed touchscreen input device and the main controller.
Depending on the choice of the main controller, the use of the MATLAB image processing toolbox could easily be integrated with Simulink, since both are MathWorks products. The Real Time Workshop toolbox could also be used to interface between the Simulink model and the main controller. Bucher and Balemi found in 2003 that “The Real-Time Workshop toolbox (RTW) generates C-code from a Simulink model without the need of any programming knowledge.”

Another important factor considered was how the vector based approach was going to be achieved. The Portrayer made by Benedettelli in 2008 gave evidence that the vectorisation process could be done using MATLAB. A similar solution would require algorithms to be developed to generate paths to be sent to the main controller. The MATLAB spline toolbox is also stated to be able to convert points to cubic splines or curves (MathWorks 2010). More details on its capabilities will become evident once research and testing has occurred.

The image processing program was developed iteratively with a combination of trial and error and MATLAB tutorials from the MathWorks website. Initially, the program development was to concentrate solely on a raster image form. The program was based on the simplified flowchart as in Figure 5.20, and the full flowcharts can be found in Appendix F.1. The program was debugged as each component of the design was added, and thoroughly tested to find possible issues which could occur during integration with the main system. The code is attached in Appendix F.2.
5.4.2 Image Transformation

Initially the image processing program reads the image file from the working directory. The initial image was represented by three different matrices which corresponded to the red, green and blue components of the image. The size of each matrix was the same as the initial resolution of the input image and hence each matrix entry corresponded to a pixel on the image. Using the image processing toolbox’s inbuilt function, the picture then underwent a transformation to a greyscale image, and finally to a single intensity matrix. At this stage, the components of the matrix were normalised to double precision – which converted the values between 0 and 1 depending on the intensity of the pixels.

Depending on the user’s choice, the image was either directly resized to the desired resolution or the aspect ratio was taken into account while resizing. This is illustrated in Figure 5.21.
Figure 5.21: Demonstration of resizing options. (a) The original image (University of Adelaide 2010). (b) Direct resizing which leads to stretching. (c) A maintained aspect ratio.

A user input value for the maximum resolution determined how detailed the image would be. Therefore, if the aspect ratio was to be kept constant, the smaller of the vertical and horizontal sizes would be scaled down to fit in the canvas. Using Figure 5.21 as an example, the maximum resolution was set to 100, which meant the vertical resolution was set to the maximum resolution, and the horizontal resolution was scaled to 85 to maintain the aspect ratio. Once all resizing transformations are completed, the next step involved the image being processed as a binary or greyscale image, depending on the user settings.

### 5.4.3 Binary Filtering

In order to construct the binary image, a low pass filter was used to filter pixels less than a user-inputted threshold value. Since all pixels of the image matrix were normalised from 0 to 1, the threshold value was also set to be between this interval. The convention used in MATLAB is reversed, in that an entry of 0 corresponds to a black pixel while a 1 entry was white. Therefore the filtering algorithm set a pixel to 0 if its grading was below the threshold value and set it to 1 otherwise. It was important to note that with a binary image, especially for one of low resolution, the amount of detail was very limited since there were only two pixel shadings, either black or white. An illustration of how threshold values affect the final image can be seen in Figure 5.22. From these images it can be seen that a threshold value of 0.25 does not include all the main features of the image, while a threshold value of 0.75 has interpreted the gold part of the picture as black. In this case the best threshold setting would be 0.5 which incorporates a suitable amount of feature from the image.
It was found that the best threshold values varied for different images and there were some images that were difficult to distinguish for any threshold value. This can be seen in Figure 5.22. This problem occurred for images which were noisy or did not have very defined edges. Consequently, the filter was found to be too coarse to effectively reproduce the desired image.
It was acknowledged that producing a binary image would produce the least detailed image type. However, this would provide a starting point to test the other components of PICARSO, such as the motors and painting module.

### 5.4.4 Greyscale Filtering

The first improvement from the binary image was incorporating the different gradients of pixels in a greyscale image. Since the program already converted the image to a greyscale form, with a value of 0 to 1 for each pixel, the painting mechanism would be able to spray for a certain time. This program could be output straight to the main controller. This would mean that, the program could convert this greyscale image to four different pixel levels, namely values of 0, 0.33, 0.66 and 1. This could be modified at a later date once the program was integrated and tested with the physical PICARSO system. The greyscale output could be compared to the original and binary images in Figure 5.24. It can be seen that the greyscale image has added detail by taking into account the different colours of the image. These images have been coloured as different shadings of grey.
5.4.5 Effect of Resolution on Conversion

In all of the previous examples, a maximum resolution of 100 pixels was used. It was found that increasing the resolution to higher values considerably improved the detail in the converted image. The effects of this can be seen in Figure 5.25 and Figure 5.26. These image’s resolutions were increased to a maximum of 300 pixels. Compared to Figure 5.23, which showed significant image noise, this resolution shows a significant improvement of the details maintained after the conversion process. However, it was known that the maximum resolution was limited to the accuracy of the motors and painting module as discussed in the design specification. Additionally, increasing the resolution led to greater computation times which was especially evident in very high resolution images (around 1000 x 1000 or more).

Figure 5.25: MATLAB output of original and binary images in higher resolution.
5.4.6 Theoretical Output Simulation

Once the image processing conversion was complete, the program was developed to show a theoretical plot of the processed image. This was done by reading the output matrix and colouring a filled in circle depending on the value of the matrix. Circles were chosen to mimic the spray gun’s painting pattern. A gradient was also added to simulate the effects of paint bleeding. Additionally, the plotting program was developed to show the effect of errors in the motor or spray gun which could affect the painted image. This was done by adding a random error between 0 and 1 for each pixel value in the matrix. This would simulate a worst case scenario. The plots for the optimal and erroneous images can be seen in Figure 5.27.
Figure 5.27: Theoretical PICARSO output plots of the University of Adelaide showing (a) optimal plotting, and (b) a plot with plot accuracy errors.

It can be seen from Figure 5.30 that positional errors have caused a significant loss of detail to the image and hence it was important to design the motor and painting mechanisms to be as precise as possible. Additionally, the simulation showed how bleeding could be used to an advantage. It was thought that a slight increase in pixel size would remove the spaces in between pixels. However, as shown in Figure 5.31, larger pixel size improved the cohesiveness of the image and would be taken into consideration in the final design.
Figure 5.28: A comparison of pixel sizes to improve image quality.

The results from the theoretical plots gave an early indication of the output capable by PICARSO. Once the full system is functional these plots could also be used to compare the actual image to theoretical image.

5.4.7 Image Processing Software Progress Evaluation

The independent image processing program was completed within the required completion time. The image was processed using MATLAB and was able to output the image in matrix form. This was verified by integrating the program into the prototype, µAngelo, which appeared to have successfully read the image output, however the prototype was not designed to physically produce the image. The program has been tested and can read RGB file formats including JPEG, PNG, BMP formats, as well as indexed formats including GIF and TIFF files. The program has not been tested on less common image file formats, however they should be compatible if readable by MATLAB.

Some research has been done in regards to the vector based extension goal. This included experimenting with the edge detection functions and spline toolbox in MATLAB. Edge detection was used to find the outlines of the images in raster form, however this still needed to be vectorised.

5.5 Control System Design

The control system will incorporate the entire software solution for the PICARSO design. The control system includes the image processing and control software. The image processing software determines the activation of each hardware component and the control software commands all of the hardware components of the system. This section will discuss the control
software architecture, functionality, motor controller selection and how the control will be realised.

5.5.1 Software Architecture

Essentially, the highest level of control acts as a finite state machine, in which desired actions can be sent to move PICARSO to desired states. The incorporation of a user friendly graphical user interface, will allow PICARSO’s actions can be controlled by a human user. At this stage, it is thought that PICARSO’s GUI will be developed using MATLAB. This was considered the most appropriate software for the GUI’s development because of the familiarity with the software and the compatibility with the existing image processing software concurrently developed in MATLAB.

The intermediate level control drives the painting module to certain positions on the working area. The exact commands of the intermediate level control software are heavily dependent on the outcomes of the image processing code. For the raster based approach, the intermediate level control will simply drive the motors to trace out a pathway covering the position of every single pixel in the working area, while controlling the spray gun solenoid to actuate depending on the image processing input.

The intermediate level control will become increasingly complex if the extension goals are considered. The implementation of greyscale images and painting using a vector based approach will significantly increase the complexity of the code. For example, using a vector based approach, regardless of the exact details of the image processing software, the program will receive either an equation defining a sequence of co-ordinates to navigate to or simply, a matrix of points defining the image.

The low level control will determine the action of the three Maxon EC45 motors. For example, when the painting module needs to travel to a desired position on the work space, the low level control will determine the exact amount of spooling of the motor is required to achieve this position.

5.5.2 Control Software

The control software is used to drive the motors to spool so that the end-effector can reach each desired position in the correct sequence to output the desired image. The control software can be divided into different levels of control. The high level functionality of the program will incorporate initialisation all of the system parameters, library and header files and then prompt the user to select the operational mode. All of the motor parameters must be set prior to
controlling the motor system. These parameters include the user specified operation mode, motor and position sensors and the controller's regulation gains. The PICARSO control system will be capable of operating in either manual or autonomous modes.

In manual mode the user is able to input a single desired position or torque value to control the motor, before being prompted for subsequent positions. Operation in manual mode allows the user to drive the painting module in single steps instead of an entire sequence of positions, which is advantageous during testing stages. Autonomous operation mode has functionality which meets the required design specifications for PICARSO. In autonomous mode the system will be able to operate independently to retrieve the input image, run image processing, calculate the inverse kinematics to determine cable lengths and drive the motors to spool to the desired lengths.

Within the aforementioned operation modes, the motor control program will be active. The motor control program receives the cable lengths required to reach the desired position and determines the angular position for each of the three servo motors. The command will be sent between the master controller and the master EPOS2 70/10 through either RS232 serial or USB connection depending on the speed required. A CAN connection exists between the three slave controllers to ensure parallel control between motors. Concurrently, the solenoid will be controlled to actuate the spray gun when required.

A graphical overview of the control software structure for this program is displayed in Appendix G.

5.5.3 Motor Controller Selection

The motor controller options that are compatible with the selected Maxon EC45 250W motors, are the EPOS (Easy-to-use Positioning) series and the second generation EPOS2 series digital positioning motor controllers. While the EPOS motors controllers are backwardly compatible, there are still advantages and disadvantages to consider when selecting the motor controller. Both motor controllers have the ability to be controlled in position, speed and current control modes and communicate via both CAN and RS232 serial interfaces. As a motor controller was required for each motor and as the EPOS2 motor controllers were more expensive, justification of additional expenditure is required. Upgrading to the EPOS2 included the possibly for USB compatibility in addition to other enhanced functionality such as Interpolated Position Mode (IPM). IPM allows the controller to synchronously drive the pathway between two points by interpolation. The potential for smoother motion and coordinated point-interpolated movement
between various axes with high accuracy may prove a significant advantage. The property of interpolated movements to position could potentially allow for smoother control of the end-effector, which depending on the control program, could provide a higher picture quality.

Ultimately, because of the wide range of benefits the EPOS2 has over the EPOS and their specific relevance to PICARSO’s application, the EPOS2 motor controller was chosen. For the required functionality of position control and to obtain the motor’s maximum output, a nominal torque of 10A is required and so the EPOS2 70/10 motor controller was selected. The specifications for motor controller are shown in Appendix D. By choosing the EPOS2 70/10, the control system can be organised by having a Master PC controlling the overall system and running each of the three EPOS2 controllers as Slaves via a CAN network as shown in Figure 5.29.

![Figure 5.29: Diagram showing relationship between master (PC) and slaves (EPOS controllers) of the motor control system (Maxon Motors 2010).](image)

The selected EPOS2 70/10 motor controller has room for up to 10 digital inputs, 2 analogue inputs and 5 digital outputs, which meet the design specifications and should be sufficient for the control of the system. This will allow for the solenoid output and extra sensor inputs.

5.5.4 Implementing Maxon Controller Functions

The Maxon motor controllers can be programmed using a variety of methods. Firstly, Maxon provides a software package with the purchase of its motors known as EPOS Studio. Using this program, the controller’s initial parameters can be found and high level commands to the motor controller can be made. For instance, using the position control mode, the position of the motor’s rotor can be moved from point to point. Furthermore, the desired torque on the motor shaft can be achieved using the current mode. EPOS Studio has been developed to remove the low level code required for control of the motor in one of the predetermined operational modes. For PICARSO, the EPOS Studio can be used in the early stages of the primary control system
development. Subsequently, the EPOS Studio can be used to test the hardware and connections, send single commands to the motor and determine controller gains through automatic tuning.

Secondly, Maxon has provided the Dynamic-link library (DLL) for the EPOS2 70/10 controllers. By providing the controller’s DLL, functions within the library can be extracted and used to retrieve information about the encoder and motor system. Furthermore, the DLL can be used to write information to the controller. By calling functions from the DLL, control can be accomplished without using Maxon EPOS Studio software. The accessibility of the controller operations allows flexibility and simplicity in the program since low level commands are not required. The Windows 32-bit DLL benefits from its ability to import EPOS2 function libraries directly into MATLAB. However, it is noted that the Windows 32-bit DLL is incompatible with real time operation.

Finally, low level programming can be implemented in the control software. At low level, each port of an RS232 serial connection (or USB if required) can be opened and commands can be sent in the required format. For example, sending a sequence of activating bits can set the motor controller operation mode or set a desired rotor position. Due to accessibility, familiarity and compatibility with the image processing code, MATLAB was selected as the programming software.

Due to flexibility and previous experience, the control program will be written in low level code and developed in MATLAB. While the EPOS Studio software provides easily accessible high level motor control through a GUI, autonomous control of the motor system using this program would be difficult. A separate program or macro would need to be developed to automate the control of the EPOS Studio program. Despite the convenience of writing the control program using EPOS2 specific functions, the DLL is unable to operate in real time. The inability to run in real time is a significant hindrance and counterproductive to meeting the design specifications. As the higher level control programming by the EPOS Studio and calling Windows 32-bit DLL files do not provide the flexibility or allow the system to meet the design specifications, a lower level control programming method must be considered.

5.5.5 Development of the PICARSO Controller

The PICARSO control system will command and regulate the behaviour of the central cable driven painting module and maintain the stability from any minor disturbances if required. In PICARSO, disturbances to the system may be unavoidable due to the movement of the spray gun and paint flow fluctuations in the air hoses. Minimising the effect of these disturbances
through control is especially important to ensure the stability of the painting module and its ability to produce painted images. The accuracy of the image output will be a measure of the system’s success.

Each motor controller is paired to one of the three motors and corresponds to that motor’s control. The selected EPOS2 70/10 motor controllers are capable of control through several operation modes, but most relevant, is the controllers’ ability to implement feedback control using position and torque. Initially, PICARSO will specify a position setpoint and compares this to the actual position of the motor shaft as determined by the Hall Effect position sensor. If there is an error between the position setpoint and the position obtained through feedback, the controller will be able to correct itself by driving the motors until the error is removed. Each motor controller is required to balance PICARSO’s painting module when the spray gun is painting the image. This is achieved by tuning each of the controllers to determine the appropriate controller gains.

5.5.6 Control Software Progress Evaluation

The control software code is still in the development stages. Currently the overall program functionality and structure has been determined, using flowcharts, with the exception of minor additions that may be implemented during testing for added safety, function and robustness of the system. These features include ‘pause’ and ‘master reset’ subroutines and checking for abnormal conditions.

The mode in which these functionalities will be realised has been selected. Low level programming has been chosen as the level of the control software and the appropriate registers to write to have been identified. The software has been developed in MATLAB.

As part of the motor selection, the Maxon EPOS2 70/10 digital motor controllers have been selected and as a result of this, the development of the controller will be based on this system. Once the controllers have been delivered, testing of the software will occur.

A theoretical controller still needs to be developed to determine the controller gains that are required to stabilise the suspended system from any undesirable rolling movement.

The control software functionality and skeleton code has been established within the required timeframe as specified in the Gantt chart. However, the development of the fully functioning final code is currently in process.
6 Costing Analysis

The group was allocated $1000 from the University to purchase components for PICARSO. An additional $3000 was received for a successful application for the Open Day Creativity and Innovation Fund for 2010. This meant a total budget of $4000 was available.

The group has received donations from both APEX and Crowie’s Paints. APEX donated a solenoid for use with the spray gun, while Crowie’s Paint has donated 30 Litres of paint for testing purposes.

The full estimated costing is denoted in Table 6.1 as well as the estimated in-kind costs in Table 6.2.
**Table 6.1** PICARSO total component costing for prototype, main system and other miscellaneous costs.

<table>
<thead>
<tr>
<th>Source</th>
<th>Unit Cost</th>
<th>Total Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Float-A-Boat</td>
<td>$45.00</td>
<td>$135.00</td>
</tr>
<tr>
<td>Bunnings</td>
<td>$30.00</td>
<td>$30.00</td>
</tr>
<tr>
<td>Anest IWATA</td>
<td>$273.00</td>
<td>$273.00</td>
</tr>
<tr>
<td>Anest IWATA</td>
<td>$114.10</td>
<td>$114.10</td>
</tr>
<tr>
<td>Anest IWATA</td>
<td>$6.93</td>
<td>$34.65</td>
</tr>
<tr>
<td>Anest IWATA</td>
<td>$4.62</td>
<td>$18.48</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$812.48</td>
<td>$2,437.44</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$702.30</td>
<td>$2,106.90</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$17.88</td>
<td>$53.64</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$26.86</td>
<td>$53.72</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$34.84</td>
<td>$34.84</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$30.10</td>
<td>$90.30</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$27.10</td>
<td>$81.30</td>
</tr>
<tr>
<td>Maxon Motors</td>
<td>$52.92</td>
<td>$158.76</td>
</tr>
<tr>
<td>To Be Decided</td>
<td>$50.00</td>
<td>$50.00</td>
</tr>
<tr>
<td>Fisherman’s Paradise</td>
<td>$50.00</td>
<td>$50.00</td>
</tr>
<tr>
<td>To Be Decided</td>
<td>$100.00</td>
<td>$100.00</td>
</tr>
<tr>
<td>$205.00</td>
<td>$440.23</td>
<td>$5,016.90</td>
</tr>
</tbody>
</table>
Table 6.2: Summary of in-kind costs.

<table>
<thead>
<tr>
<th>In-Kind Costs</th>
<th>#</th>
<th>Source</th>
<th>Estimated Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solenoid</td>
<td>1</td>
<td>APEX Robotics and Automation Pty. Ltd.</td>
<td>$50</td>
</tr>
<tr>
<td>Paint</td>
<td>30 L</td>
<td>Crowie's Paint - Malvern</td>
<td>$400</td>
</tr>
</tbody>
</table>

The additional costs to the project were in the form of labour costs. These were evaluated based on the hours spent by all group members while working on the project at standard labour rates as seen in Table 6.3. A standard hourly rate of $26 per hour was used, which equates to an annual salary of approximately $50,000. At this point, the project is approximately half way through the project time line and hence the total salary is at an expected value of $20,820.

Table 6.3: Total hour log for January – April 2010 and associated labour costs.

<table>
<thead>
<tr>
<th>Annual Salary</th>
<th>$50,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hourly Rate</td>
<td>$26</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Month</th>
<th>Sven</th>
<th>Neil</th>
<th>Sam</th>
<th>Joyce</th>
<th>Ian</th>
<th>Monthly Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>January</td>
<td>21</td>
<td>19</td>
<td>10</td>
<td>20</td>
<td>20</td>
<td>90</td>
</tr>
<tr>
<td>February</td>
<td>19</td>
<td>17</td>
<td>10</td>
<td>20</td>
<td>24</td>
<td>90</td>
</tr>
<tr>
<td>March</td>
<td>86</td>
<td>44</td>
<td>28</td>
<td>60</td>
<td>57</td>
<td>275</td>
</tr>
<tr>
<td>April</td>
<td>91</td>
<td>54</td>
<td>72</td>
<td>71</td>
<td>69</td>
<td>357</td>
</tr>
</tbody>
</table>

Total hours as of end of April 2010: 812
Total Salary: $20,820.51
7 Future Work

This section discusses the work remaining to be completed in order to successfully complete the project. The remaining work will be completed in accordance with the project Gantt chart (see Appendix A). Additionally, it is intended to have PICARSO built and functional by The University of Adelaide Open Day on the 15th August 2010, for the purpose of displaying the system and the achievements of the student engineers undertaking this project.

7.1 Hardware System

The PICARSO system will be assembled when the custom components are manufactured. This is scheduled for July 2010. Testing and debugging of the control systems as well as testing of the general operation of the assembled rig will occur. The wooden mounting frame will be constructed after small scale testing is completed. The cables will be specified when the painting module components are assembled. Additional testing will occur in order to determine the ideal cable attachment to the end-effector as well as the configuration and separation of the cables. Furthermore aesthetic adjustments to the wall mounts will be considered, such as Perspex covers. Finally, the electrical wiring will be purchased to cater for both 3.0×3.0m and 5.0×5.0m working areas.

7.2 Painting System

To complete the full configuration of the system, the remaining parts needed are additional connections between painting components, a suitable air compressor and air regulator. The system needs to be tested to identify suitable spray operation distances, the amount of paint expelled, and the spray pattern sizes which are achievable for the painting module. Furthermore, the painting housing must be manufactured and attached to the connecting wires from the servo motors, and set up for suitable full scale system testing.

A forth design for the painting housing has been proposed to include functionality to allow the painting module to rotate around the centre of rotation with respect to the spray gun nozzle. This design will be finalised before the end of the semester break to begin manufacturing as soon as possible.

7.3 Image Processing Software

The image processing component of the design has been mostly completed, but the success of the program will not be known until it can be integrated with the full scale controller. Until then, it is proposed that a GUI is to be developed using MATLAB's GUI Design Environment.
GUI will include options to set the desired maximum resolution, threshold, filter type and so on. This will be developed to be user friendly, so that it can be used with the full scale system if required. The theoretical output plot, as in Figure 5.27, will also be incorporated into the GUI, to give the user a visual representation of the proposed painted image. Time permitting, a cropping function may be utilised to allow the user to crop their image to a desired shape.

Further work in image processing will include the development of a vector based approach using MATLAB or otherwise. The edges of the picture have been found using MATLAB’s edge function, however the vectorisation process still needs to be completed. Research has been done to produce a path finding algorithm similar to one used by the Portrayer (Benedettelli 2008). This will use an iterative procedure to link neighbouring points into chains to produce a vector as described in Erik’s XY-Plotter in 2007. This is likely to be attempted using MATLAB.

Depending on time constraints, the image processing program will also be developed to integrate with a touch screen interface through Simulink or otherwise. There has been no research as of yet as to how this can be achieved, however based on current knowledge it is proposed that Simulink’s input device blocks could be used in this case.

### 7.4 Control Software

One of the functional requirements for the PICARSO system was scalability of up to 5.0×5.0m. Implementing scalability within the control software still needs to be considered to meet this requirement. One of the major future works is the method the system detects the size of the canvas. The size of the canvas must be identified prior to operation, as this defines the operational area of the system. This area will be used in the program when scaling an image to fit the canvas. Additionally, to allow for scalability and automation of the system, the maximum motor spooling distances, parameters of the motors, and controller gains will need to be determined. At this stage, the exact method of detecting the working area has not been verified and the program is still to be written.

The most successful theoretical controller is still to be determined but several established techniques will be used to calculate the most appropriate controller transfer function. Once the theoretical controller has been determined, a model of the system can be developed in Simulink and the full scaled PICARSO system has been assembled, the theoretical controller gains can be tested and the resultant effect on the motors can be simulated.
To meet the design specification of a scalable system, the controller gains need to be automatically tuned within the software. A program to achieve this specification is still to be developed.
References


Bucher, R & Balemi, S 2006, ‘Rapid controller prototyping with Matlab/Simulink and Linux, Control Engineering Practice,’ *Special Section on Advances in Control Education*, vol 14, no. 2, pp. 185-192.


Maxon Motors Australia 2010, *EPOS2 70/10 datasheets and manuals*, viewed 10th April

Maxon Motors Australia 2010, *EC45 250W motor datasheet*, viewed 10th April
<http://shop.maxonmotor.com/ishop/download/article/136209.xml>


RoboPainter Maryland 2008, *RoboPainter Maryland*, Maryland, viewed 3 January 2010,

RoboPainter Texas 2008, *RoboPainter Texas*, Texas, viewed 3 January 2010,

Spiritus-Temporis 2005, *Venturi Effect, USA*, viewed 19th April 2010,
<http://www.spiritus-temporis.com/venturi-effect/>


Figure 2.1: Lehni, J & Franke, U 2002, *Hektor*, viewed 29 December 2009, p. 45

Figure 2.2: Lehni, J & Franke, U 2002, *Hektor*, viewed 29 December 2009, p. 29

Figure 2.3: Benedettelli, D 2008, *NXT Portrayer Robot*, viewed 11th April 2010
<http://robotics.benedettelli.com/portrayer.htm>

Figure 2.4: RoboPainter Texas 2008, *RoboPainter Texas*, Texas, viewed 3 January 2010,

Figure 2.5: Ottaviano, E, Ceccarelli, M & Pelagalli, P 2006, ‘A performance analysis of a 4 cable-driven parallel manipulator’, *IEEE Xplore Digital Library*, viewed 12 March 2010, p. 2

Figure 2.6: Ottaviano, E, Ceccarelli, M & Pelagalli, P 2006, ‘A performance analysis of a 4 cable-driven parallel manipulator’, *IEEE Xplore Digital Library*, viewed 12 March 2010, p. 1

Figure 5.1: Wiley Printing 2008, *Raster Vs. Vector Art*, viewed 10th May 2010 <http://www.wileyprinting.com/images/arthelp/rastervector.gif>

Figure 5.6, Figure 5.7, Figure 5.29, Figure D.1, Figure D.2, Figure D.3, Figure D.4, Figure D.5, Figure D.6, Figure D.7, Figure D.8: Maxon Motors Australia 2010, *EPOS2 70/10 datasheets and manuals*, viewed 10th April, <http://www.maxonmotor.com.au/downloads.asp>.


Figure 5.10: China Suppliers 2010, *Pressure-fed Spray Gun System*, China Suppliers, viewed 19th of April, <http://image.made-in-china.com/4f0j00VMtaTflBYoH/Paint-Tank-W-Spray-Gun-RP8313-.jpg>

Figure 5.11: Way Builder 2010, *Pressure-fed Spray Gun*, Way Builder Network, viewed 19th of April <http://www.sweethaven02.com/BldgConst/Painting01/painti70.gif>

Figure 5.12, Figure 5.13: Anest Iwata 2010, *SGA101 Automatic Spray Gun*, Anest Iwata, viewed 2nd April <http://www.anest-iwata.com.au/range/img/products/thumbs/T-sga101.jpg>


Figure B.2: PicoBytes Inc 2005, *PicoPic user & technical manual*, PicoBytes Inc. p. 2


Figure E.2, Figure E.3: Grex 2010, *Genisis XBi – Siphon-fed Airbrush*, Grex Power Tools, viewed 12th April 2010, <http://www.grexusa.com/grexairbrush/pics/m_Genesis.XBi_full.jpg>
Appendix

A Gantt Chart

The following section shows a scaled version of the Gantt chart which shows the timeline of the PICARSO project.
B  μAngelo

The following section contains information regarding the prototype development.

B.1 Motor Specifications

The motor specifications for the Hitec HS-785HB Sail Winch Servo used in the μAngelo system are detailed in this section from Lee (2003) and Hitec/Multiplex USA (2003).

- Motor Type: 3 pole
- Control: Pulse Width Modulation (PWM), 900µs - 2100µs, 1500µs (Neutral)
- Operating Voltage: 4.8V – 6.0V
- Operating Speed (No Load): 1.68s/360° @ 4.8V, 1.38 s/360° @ 6.0V
- Stall Torque: 11kgcm @ 4.8V, 13.2kgcm @ 6.0V
- Standing Torque: 8.8kgcm @ 4.8V, 10.5kgcm @ 6.0V
- Running Current (No Load): 230mA/60° @ 4.8V, 250mA/60° @ 6.0V
- Stall Current: 1500mA @ 4.8V, 1800mA @ 6.0V
- Winch Travel: ± 3.5 turns from neutral
- Dimensions: 59×29×50mm
- Weight: 110g

Figure B.1: Hitec HS-785HB Sail Winch Servo (Hitec/Multiplex USA 2003)
B.2 PicoPic Specifications

The technical specifications for the PicoPic controller used in the µAngelo system are listed in the following section.

- Processor: PIC18F242
- Operation Frequency: 16MHz
- Maximum Number of Servos: 20
- Operating Voltage: 5V (± 5%)
- Current Draw: ≤ 14mA
- Power Consumption: 70mW
- Asynchronous Serial Port: RS232 (9600 default Baud Rate)
- Dimensions (approx.): 38.3×63.6mm
- Weight: 13.7g
- Servo Pulse Width Range: 500µs - 2500 µs

Figure B.2: PicoPic (PicoBytes Inc 2005, p. 2)
B.3 Control Software Flow Diagrams

The following flowcharts were developed to aid in the writing of the µAngelo control software.

Figure B.3: Control software flow diagrams. (a) Point-to-point control software. (b) Point-to-point control with trajectory path interpolation software.
B.4 Control Software Code

The following section shows the MATLAB code which was used to control μAngelo.

Point-to-Point Control Software

```matlab
%% SCALED SYSTEM (μAngelo) PICOPIC CONTROL - Tri-Motor User Defined Trajectories

% Date Created:                 09/04/2010
% Author:                       Sven Paschburg

% Modifications:                Name                Date
% ~ Kinematics Updated         Sven Paschburg      19/04/2010
% ~ Comments Updated           Sven Paschburg      04/05/2010

% Purpose: This code provides the basic RS232 PicoPic control functionality for μAngelo (the PICARSO scaled system).
% Key functionalities include:
%       - Tri-motor configuration control
%       - User inputs for end-effector coordinate point commands
%       - User Defined Trajectory Points
%       - Point-to-point end-effector control

%% Serial Port Initialisation
clc % clear the command window
while 1
    disp('')
    disp('Do you want to initialise the serial port?')
    reply = input('
Enter "y" if running for the first time or press ENTER to continue: ', 's');
    if  reply == 'y'
        % !!!Note: This command must be run before you run this program for the first time
        s = serial('com1','baud',9600,'timeout',1)
        break
    elseif isempty(reply)
        break
    else
        disp('')
        disp('INVALID input')
    end
end  % END while 1
fopen(s)
disp('Serial port opened')

%% Constants & Parameters
% End-Effector Parameters:
% ~ Upper Cables End-Effector Centre Offset (mm):
A = 14;
% ~ End-Effector Half-Height (mm):
B = 12.5;

% Canvas Parameters:
```
% ~ Half-Width (mm):
W = 670/2 - A; % = 321 (Max Value)
% W = 150;
% ~ Height (mm):
H = 670 - 2*B; % = 645 (Max Value)
% H = 150;

% Trajectory Parameters:
% ~ Interpolation Resolution (mm):
res = 5;
% ~ Pause time between each interpolated point (seconds):
pt = 0.5;

% Servo Parameters:
% ~ Winch Drum Diameter (mm):
D = 38;

%% End-Effector Initialisation
% Set the end-effector to the starting point of the canvas (0,0)
X = 0;
Y = 0;

L1 = (X^2 + (Y - B + ((670 - H)/2))^2)^(1/2);
L2 = ((W - X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^(1/2);
L3 = ((W + X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^(1/2);

t1 = L1/pi/D;
t2 = L2/pi/D;
t3 = L3/pi/D;

p1 = 2500 - 226*t1;
p2 = 2500 - 222*t2;
p3 = 2500 - 225*t3;

fwrite(s,[120 10 floor(p1/256) floor(mod(p1,256)) 0])
fwrite(s,[120 1 floor(p2/256) floor(mod(p2,256)) 0])
fwrite(s,[120 20 floor(p3/256) floor(mod(p3,256)) 0])

%% Trajectory Definition
while 1
disp('')
disp('Press ENTER without an input value to exit')
disp('or enter a valid coordinate to continue')

X = input('
Enter X Coordinate (mm): ');
if isempty(X)
    break
end
while (X>W) || (X<-W)
    X = input('
X Coordinate is INVALID, please enter a valid value: ');
end

Y = input('
Enter Y Coordinate (mm): ');
if isempty(Y)
    break
end
end
while (Y>H)||(Y<0)
    Y = input('nY Coordinate is INVALID, please enter a valid value:
');
end

%% Inverse Kinematics
% Cable Lengths:
% !!!Note: These lengths do not take into account the spring lengths
L1 = (X^2 + (Y - B + ((670 - H)/2))^2)^1/2;
L2 = ((W - X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^1/2;
L3 = ((W + X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^1/2;

% Number of turns of the drum (from fully wound up):
t1 = L1/pi/D;
t2 = L2/pi/D;
t3 = L3/pi/D;

%% Servo Commands

% Period Offset (microseconds) :
% (2500 = fully wound up)
p1 = 2500 - 226*t1;
p2 = 2500 - 222*t2;
p3 = 2500 - 225*t3;

% PicoPic Commands:
% ~ Format:
% [Address , Channel , Position High Byte , Position Low Byte , Speed]

~ Motor 1:
fwrite(s,[120 10 floor(p1/256) floor(mod(p1,256)) 0])
~ Motor 2:
fwrite(s,[120 1 floor(p2/256) floor(mod(p2,256)) 0])
~ Motor 3:
fwrite(s,[120 20 floor(p3/256) floor(mod(p3,256)) 0])

end % END while 1

%% End-Effector Reset
% Set the end-effector to the origin point (0,0)
X = 0;
Y = 0;

L1 = (X^2 + (Y - B + ((670 - H)/2))^2)^1/2;
L2 = ((W - X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^1/2;
L3 = ((W + X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^1/2;
t1 = L1/pi/D;
t2 = L2/pi/D;
t3 = L3/pi/D;
p1 = 2500 - 226*t1;
p2 = 2500 - 222*t2;
p3 = 2500 – 225*t3;

fwrite(s,[120 10 floor(p1/256) floor(mod(p1,256)) 0])
fwrite(s,[120 1 floor(p2/256) floor(mod(p2,256)) 0])
fwrite(s,[120 20 floor(p3/256) floor(mod(p3,256)) 0])

%%% Serial Port Shutdown
fclose(s)
disp('')
disp('Serial port closed')
disp('')
disp('PROGRAM ENDED')

Point-to-Point Control with Trajectory Path Interpolation Software

%%% SCALED SYSTEM PICOPIC CONTROL - Trajectory Straight Line Interpolation

% Date Created: 09/04/2010
% Author: Sven Paschburg

% Modifications:
% ~ Kinematics Updated        Sven Paschburg    19/04/2010
% ~ Comments Updated          Sven Paschburg    04/05/2010

% Purpose: This code provides the basic RS232 PicoPic control functionality
% for uAngelo (the PICARSO scaled system).
% Key functionalities include:
%   - Tri-motor configuration control
%   - User inputs for end-effector coordinate point commands
%   - Point-to-point end-effector control with linear
%     interpolation of the trajectory path

%%% Serial Port Initialisation
clc % clear the command window

while 1
    disp('')
    disp('Do you want to initialise the serial port?')
    reply = input('Enter "y" if running for the first time or press ENTER
to continue: ', 's');
    if reply == 'y'
        % !!!Note: This command must be run before you run this program for
        % the
        % first time
        s = serial('com1','baud',9600,'timeout',1)
        break
    elseif isempty(reply)
        break
    else
        disp('')
        disp('INVALID input')
    end
end

fopen(s)
disp('Serial port opened')

%%% Constants & Parameters
% End-Effector Parameters:
% ~ Upper Cables End-Effector Centre Offset (mm):
A = 14;
% ~ End-Effector Half-Height (mm):
B = 12.5;

% Canvas Parameters:
% ~ Half-Width (mm):
W = 670/2 - A; % = 321 (Max Value)
% W = 150;
% ~ Height (mm):
H = 670 - 2*B; % = 645 (Max Value)
% H = 150;

% Trajectory Parameters:
% ~ Interpolation Resolution (mm):
res = 5;
% ~ Pause time between each interpolated point (seconds):
pt = 0.25;

% Servo Parameters:
% ~ Winch Drum Diameter (mm):
D = 38;

%% End-Effector Initialisation
% Set the end-effector to the starting point of the canvas (0,0)

X = 0;
Y = 0;

L1 = (X^2 + (Y - B + ((670 - H)/2))^2)^(1/2);
L2 = ((W - X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^(1/2);
L3 = ((W + X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^(1/2);

t1 = L1/pi/D;
t2 = L2/pi/D;
t3 = L3/pi/D;

p1 = 2500 - 226*t1;
p2 = 2500 - 222*t2;
p3 = 2500 - 225*t3;

fwrite(s,[120 10 floor(p1/256) floor(mod(p1,256)) 0])
fwrite(s,[120 1 floor(p2/256) floor(mod(p2,256)) 0])
fwrite(s,[120 20 floor(p3/256) floor(mod(p3,256)) 0])

%% Trajectory Definition
% Initialise the coordinate points:
X1 = 0;
Y1 = 0;

while 1
    disp(''
    disp('Press ENTER without an input value to exit'
    disp('or enter a valid coordinate to continue'

    X2 = input('
Enter X Coordinate (mm): ');
if isempty(X2)
    break
end
while (X2>W) || (X2<-W)
    X2 = input('\nX Coordinate is INVALID, please enter a valid value: ');
end

Y2 = input('\nEnter Y Coordinate (mm): ');
if isempty(Y2)
    break
end
while (Y2>H) || (Y2<0)
    Y2 = input('\nY Coordinate is INVALID, please enter a valid value: ');
end

dX = X2 - X1;
dY = Y2 - Y1;

% Set the next coordinate point:
X1 = X2;
Y1 = Y2;

if dX == 0
    X = X2;
end
if dY == 0
    Y = Y2;
end

while (X ~= X2) || (Y ~= Y2)
    \% Trajectory Interpolation
    if X ~= X2
        if dX > 0
            X = X + res;
        elseif dX < 0
            X = X - res;
        end
    end
    if Y ~= Y2
        if dY > 0
            Y = Y + res;
        elseif dY < 0
            Y = Y - res;
        end
    end

    \% Inverse Kinematics
    \% Cable Lengths:
    \% !!!Note: These lengths do not take into account the spring lengths
    L1 = (X^2 + (Y - B + ((670 - H)/2))^2)^(1/2);
    L2 = ((W - X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 -
    H)/2))^2)^(1/2);
    L3 = ((W + X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 -
    H)/2))^2)^(1/2);

% Number of turns of the drum (from fully wound up):
t1 = L1/pi/D;
t2 = L2/pi/D;
t3 = L3/pi/D;

%%% Servo Commands

% Period Offset (microseconds):
% (2500 = fully wound up)
p1 = 2500 - 226*t1;
p2 = 2500 - 222*t2;
p3 = 2500 - 225*t3;

%%% PicoPic Commands:
% ~ Format:
% [Address , Channel , Position High Byte , Position Low Byte , Speed]

% ~ Motor 1:
fwrite(s,[120 10 floor(p1/256) floor(mod(p1,256)) 0])
% ~ Motor 2:
fwrite(s,[120 1 floor(p2/256) floor(mod(p2,256)) 0])
% ~ Motor 3:
fwrite(s,[120 20 floor(p3/256) floor(mod(p3,256)) 0])

pause(pt);
end % END while (X~=X2)&&(Y~=Y2)
end % END while 1

%%% End-Effector Reset
% Set the end-effector to the origin point (0,0)

X = 0;
Y = 0;

L1 = (X^2 + (Y - B + ((670 - H)/2))^2)^(1/2);
L2 = ((W - X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^(1/2);
L3 = ((W + X - A + ((670 - 2*W)/2))^2 + (H - Y - B + ((670 - H)/2))^2)^(1/2);

t1 = L1/pi/D;
t2 = L2/pi/D;
t3 = L3/pi/D;

p1 = 2500 - 226*t1;
p2 = 2500 - 222*t2;
p3 = 2500 - 225*t3;

fwrite(s,[120 10 floor(p1/256) floor(mod(p1,256)) 0])
fwrite(s,[120 1 floor(p2/256) floor(mod(p2,256)) 0])
fwrite(s,[120 20 floor(p3/256) floor(mod(p3,256)) 0])

%%% Serial Port Shutdown
fclose(s)
disp('')
disp('Serial port closed')
disp('')
disp('PROGRAM ENDED')
B.5 Design Calculations

The derivations for the maximum spool length, reference frame dimensions and inverse kinematics, as well as the required motor turns and motor positions values are shown in this section. These calculations depend on several parameters, which are defined as follows:

- $T_{\text{max}}$ – Maximum winch travel
- $D$ – Diameter of the winch drum
- $L_{\text{max}}$ – Maximum spool length
- $L_{\text{ref}}$ – Side length of the square reference frame
- $L_1$ – Length of cable 1
- $L_2$ – Length of cable 2
- $L_3$ – Length of cable 3
- $W$ - Half-width of the canvas
- $H$ – Height of the canvas
- $A$ – Upper-cables end-effector centre offset
- $B$ – End-effector half-height
- $x$ – $x$-coordinate of end-effector position
- $y$ – $y$-coordinate of end-effector position
- $T_1$ – Number of motor 1 turns from fully wound up
- $T_2$ – Number of motor 2 turns from fully wound up
- $T_3$ – Number of motor 3 turns from fully wound up
- $P_1$ – Motor 1 position period
- $P_2$ – Motor 2 position period
- $P_3$ – Motor 3 position period

The maximum spool length and side length of the square reference frame can be found using the following method.

The maximum number of turns of the motor and motor winch diameter are:

\[
T_{\text{max}} = 8 \text{ turn} \quad \text{(B.1)}
\]
\[
D = 38\text{mm} \quad \text{(B.2)}
\]

From these values and utilising Equation B.3, the maximum spool length of the motors can be calculated.

\[
L_{\text{max}} = \pi DT_{\text{max}} \quad \text{(B.3)}
\]

- 100 -
From the maximum spool length, the required reference frame dimension can be calculated.

\[
L_{\text{max}}^2 = 2L_{\text{ref}}^2
\]  
(B.4).

\[
955.0^2 = 2L_{\text{ref}}^2
\]
\[
L_{\text{ref}}^2 = \frac{955.0^2}{2}
\]
\[
L_{\text{ref}} = 675.3
\]
\[
L_{\text{ref}} \approx 670\text{mm}
\]

From the result above, we round down the value as the calculation is for a maximum length, therefore the side length of the square reference frame was approximated to 670mm.

The required cable lengths for a given end-effector (x,y) position can now be calculated using basic geometry.

Using Pythagoras' theorem, we can calculate the length for cable 1 where,

\[
L_1 = \left[ x^2 + \left( y - B + \frac{670-H}{2} \right)^2 \right]^{\frac{1}{2}}
\]  
(B.6)

and for cable 2 where,

\[
L_2 = \left[ (W - x - A + \frac{670-2W}{2})^2 + (H - y - B + \frac{670-H}{2})^2 \right]^{\frac{1}{2}},
\]  
(B.7)

and then for cable 3 where,

\[
L_3 = \left[ (W + x - A + \frac{670-2W}{2})^2 + (H - y - B + \frac{670-H}{2})^2 \right]^{\frac{1}{2}},
\]  
(B.8)

Using Equations B.6, B.8 and B.10, the required number of turns of cable that each motor has to spool off from a fully wound up initial position was calculated.

The number of turns required for motor 1 is,
while the number of turns required for motor 2 is,

\[ T_2 = \frac{L_2 \times D}{\pi}, \quad (B.12) \]

and the number of turns required for motor 3 is,

\[ T_3 = \frac{L_3 \times D}{\pi}. \quad (B.13) \]

Now using Equations B.11, B.12 and B.13, the required positions of each motor can be calculated, which at a lower level, is realised by sending a certain period value to the motors for a given position. A period of 2500\(\mu\)s corresponds to the motors being fully wound up.

The period for motor 1 is,

\[ P_1 = 2500 - 226 \times T_1, \quad (B.14) \]

while the period for motor 2 is,

\[ P_2 = 2500 - 222 \times T_2, \quad (B.15) \]

and the period for motor 3 is,

\[ P_3 = 2500 - 225 \times T_3. \quad (B.16) \]

The resulting values obtained from Equations B.14, B.15 and B.16, were sent in the appropriate format to the PicoPic, which then commanded the respective motors to their desired positions. The values that each motor turn \((T)\) is pre-multiplied by, in the equations above, represent the average period decrement to spin the motors one complete revolution. These values were obtained through a trial and error approach by incrementing the current period by a certain value until one exact revolution was achieved and then repeated at different points in the motor period range. Testing showed the motor period range was 600\(\mu\)s to 2500\(\mu\)s, which provided the eight full revolutions.
C Mechanical System Custom Components

The following section contains the detailed drawings of the hardware system including the motor mounts and spool and spray gun housing.
APPENDIX

PICARSO: 1028
Upper Angle Bracket
Al. Angle 100x50x6mm

All Dimensions in mm unless otherwise stated

Do not scale

Qty 2 off mirror image
Debur and break all edges

<table>
<thead>
<tr>
<th>Dim</th>
<th>Tolerance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5 &lt; 3</td>
<td>±0.05</td>
</tr>
<tr>
<td>3 &lt; 6</td>
<td>±0.05</td>
</tr>
<tr>
<td>6 &lt; 30</td>
<td>±0.1</td>
</tr>
<tr>
<td>30 &lt; 120</td>
<td>±0.15</td>
</tr>
<tr>
<td>120 &lt; 400</td>
<td>±0.2</td>
</tr>
<tr>
<td>400 &lt; 1000</td>
<td>±0.3</td>
</tr>
<tr>
<td>1000 &lt; 2000</td>
<td>±0.5</td>
</tr>
<tr>
<td>2000 &lt; 4000</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Check: 14/05/13
Printed: 14/05/13
Date: 14/05/13

Scale 1:1
Sheet A3
Part Number 003
Quantity 02

1 of 1
Debur and break all edges

Do not scale

PICARSO: 1028
Bearing Spacer
Al, 86×26×8mm
Debur and break all edges

Do not scale
Debur and break all edges
D  Motors and Motor Components

The following section contains datasheets corresponding to the motors and their associated components, which were chosen to be used for PICARSO.

Figure D.1: Maxon Motor EC45 250W motor data sheet (Maxon Motors Australia 2010).
Figure D.3: Maxon Motor HEDL 9140 encoder data sheet (Maxon Motors Australia 2010).
APPENDIX

Mechanical Data

<table>
<thead>
<tr>
<th>Mechanical Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight</td>
<td>approx. 330 g</td>
</tr>
<tr>
<td>Dimensions (L x W x H)</td>
<td>150 x 93 x 27 mm</td>
</tr>
<tr>
<td>Mounting plate</td>
<td>for M3 screws</td>
</tr>
</tbody>
</table>

Table 3-11 Mechanical Data

Figure D.4: Maxon Motor EPOS2 70/10 motor controller mechanical data (Maxon Motors Australia 2010).

Electrical Data

<table>
<thead>
<tr>
<th>Rating</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal power supply voltage $V_{DC}$</td>
<td>11...70 VDC</td>
</tr>
<tr>
<td>Nominal logic supply voltage $V_L$ (optional)</td>
<td>11...70 VDC</td>
</tr>
<tr>
<td>Absolute minimum supply voltage</td>
<td>10 VDC</td>
</tr>
<tr>
<td>Absolute max. supply voltage</td>
<td>75 VDC</td>
</tr>
<tr>
<td>Max. output voltage</td>
<td>$0.9 \times V_{DC}$</td>
</tr>
<tr>
<td>Max. output current $I_{max}$ (+1sec)</td>
<td>25 A</td>
</tr>
<tr>
<td>Continuous output current $I_{cont}$</td>
<td>10 A</td>
</tr>
<tr>
<td>Switching frequency</td>
<td>50 kHz</td>
</tr>
<tr>
<td>Max. efficiency</td>
<td>94%</td>
</tr>
<tr>
<td>Sample rate Pi – current controller</td>
<td>10 kHz</td>
</tr>
<tr>
<td>Sample rate Pi – speed controller</td>
<td>1 kHz</td>
</tr>
<tr>
<td>Sample rate PID – positioning controller</td>
<td>1 kHz</td>
</tr>
<tr>
<td>Max. speed @ sinusoidal commutation (motors with 1 pole pair)</td>
<td>25 000 rpm</td>
</tr>
<tr>
<td>Max. speed @ block commutation (motors with 1 pole pair)</td>
<td>100 000 rpm</td>
</tr>
<tr>
<td>Built-in motor choke per phase</td>
<td>25 µH / 10 A</td>
</tr>
</tbody>
</table>

Table 3-3 Electrical Data – Rating

Figure D.5: Maxon Motor EPOS2 70/10 motor controller electrical data (Maxon Motors Australia 2010).
## APPENDIX

### Inputs

<table>
<thead>
<tr>
<th>Input Description</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hall sensor signals</td>
<td>Hall sensor 1, Hall sensor 2 and Hall sensor 3 for Hall effect sensor ICs (Schmitt trigger with open collector output)</td>
</tr>
<tr>
<td>Encoder signals</td>
<td>A, A, B, B, I, II (max. 5 MHz)</td>
</tr>
<tr>
<td>Digital Input 1 (&quot;General Purpose&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Digital Input 2 (&quot;General Purpose&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Digital Input 3 (&quot;General Purpose&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Digital Input 4 (&quot;Home Switch&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Digital Input 5 (&quot;Positive Limit Switch&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Digital Input 6 (&quot;Negative Limit Switch&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Digital Input 7 (&quot;High Speed Command&quot;)</td>
<td>Internal line receiver EIA RS422 Standard or (Sin/Cos input), resolution 12-bit, ±1.8 V (differential)</td>
</tr>
<tr>
<td>Digital Input 8 (&quot;High Speed Command&quot;)</td>
<td>Internal line receiver EIA RS422 Standard or (Sin/Cos input), resolution 12-bit, ±1.8 V (differential)</td>
</tr>
<tr>
<td>Digital Input 9 (&quot;High Speed Command&quot;)</td>
<td>Internal line receiver EIA RS422 Standard</td>
</tr>
<tr>
<td>Digital Input 11 (&quot;Power Stage Enable&quot;), optically isolated</td>
<td>+9..+24 VDC (RI = 1.8 kΩ)</td>
</tr>
<tr>
<td>Analog Input 1</td>
<td>Resolution 12-bit 0..+5 V (differential)</td>
</tr>
<tr>
<td>Analog Input 2</td>
<td>Resolution 12-bit 0..+5 V (differential)</td>
</tr>
<tr>
<td>+V Opto IN</td>
<td>+12..-24 VDC</td>
</tr>
<tr>
<td>CAN ID (CAN Identification)</td>
<td>ID 1..127 configurable via DIP switch or software</td>
</tr>
</tbody>
</table>

**Figure D.6:** Maxon Motor EPOS2 70/10 motor controller input data (Maxon Motors Australia 2010).

### Outputs

<table>
<thead>
<tr>
<th>Output Description</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital Output 1 (&quot;General Purpose&quot;), optically isolated</td>
<td>max. 24 VDC (I &lt; 20 mA)</td>
</tr>
<tr>
<td>Digital Output 2 (&quot;General Purpose&quot;), optically isolated</td>
<td>max. 24 VDC (I &lt; 20 mA)</td>
</tr>
<tr>
<td>Digital Output 3 (&quot;General Purpose&quot;), optically isolated</td>
<td>max. 24 VDC (I &lt; 20 mA)</td>
</tr>
<tr>
<td>Digital Output 4 (&quot;Brake&quot;), optically isolated</td>
<td>max. 24 VDC (I &lt; 500 mA)</td>
</tr>
<tr>
<td>Digital Output 5 (&quot;High Speed Output&quot;)</td>
<td>Internal line driver EIA RS422 Standard</td>
</tr>
</tbody>
</table>

**Table 3-5** Electrical Data – Outputs

### Voltage Outputs

<table>
<thead>
<tr>
<th>Voltage Description</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Encoder supply voltage</td>
<td>+5 VDC (I &lt; 100 mA)</td>
</tr>
<tr>
<td>Hall sensors supply voltage</td>
<td>+5 VDC (I &lt; 50 mA)</td>
</tr>
<tr>
<td>Auxiliary output voltage</td>
<td>+5 VDC (I &lt; 150 mA)</td>
</tr>
<tr>
<td>Reference output voltage</td>
<td>+5 VDC (R = 1 kΩ)</td>
</tr>
</tbody>
</table>

**Table 3-6** Electrical Data – Voltage Outputs

### Motor Connections

<table>
<thead>
<tr>
<th>Motor Description</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>maxon EC motor</td>
<td>maxon DC motor</td>
</tr>
<tr>
<td>Motor winding 1</td>
<td>+ Motor</td>
</tr>
<tr>
<td>Motor winding 2</td>
<td>- Motor</td>
</tr>
<tr>
<td>Motor winding 3</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3-7** Electrical Data – Motor Connections

**Figure D.7:** Maxon Motor EPOS2 70/10 motor controller output data (Maxon Motors Australia 2010).
### Interfaces

<table>
<thead>
<tr>
<th>Interface</th>
<th>Signal</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS232</td>
<td>RxD; TxD</td>
<td>max. 115200 bps</td>
</tr>
<tr>
<td>USB 2.0 (full speed)</td>
<td>Data+; Data-</td>
<td>max. 12 Mbit/s</td>
</tr>
<tr>
<td>CAN 1</td>
<td>CAN_H (high); CAN_L (low)</td>
<td>max. 1 Mbit/s</td>
</tr>
<tr>
<td>CAN 2</td>
<td>CAN_H (high); CAN_L (low)</td>
<td>max. 1 Mbit/s</td>
</tr>
</tbody>
</table>

Table 3-8 Electrical Data – Interfaces

### Status Indicators

<table>
<thead>
<tr>
<th>Operation</th>
<th>Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>green LED</td>
</tr>
<tr>
<td>Error</td>
<td>red LED</td>
</tr>
</tbody>
</table>

Table 3-9 Electrical Data – LEDs

### Connections

<table>
<thead>
<tr>
<th>Component</th>
<th>On board</th>
<th>Suitable plug</th>
<th>Suitable terminal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power Supply</td>
<td>dual row male header (2 poles)</td>
<td>suitable plug</td>
<td>scotchlok terminal</td>
</tr>
<tr>
<td>Logic Supply</td>
<td>dual row female receptacle (2 poles)</td>
<td>suitable plug</td>
<td>scotchlok terminal</td>
</tr>
<tr>
<td>Motor</td>
<td>dual row male header (4 poles)</td>
<td>suitable plug</td>
<td>scotchlok terminal</td>
</tr>
<tr>
<td>Hall</td>
<td>dual row female receptacle (6 poles)</td>
<td>suitable plug</td>
<td>scotchlok terminal</td>
</tr>
<tr>
<td>Encoder</td>
<td>Plug DIN41651 (10 poles)</td>
<td>suitable plug</td>
<td>C42334-A421-C42 (female)</td>
</tr>
</tbody>
</table>

Table 3-10 Electrical Data – Connections

Figure D.8: Maxon Motor EPOS2 70/10 motor controller miscellaneous data (Maxon Motors Australia 2010).
E Sprayer Gun Components

The following section outlines additional information on different aspects of the painting systems of PICARSO. Included are details of airbrush options considered for the project, example pictures of different types of airbrushes and spray guns, a detailed parts list for the chosen spray gun, and the initial CAD design for the SGA-101 spray gun.

E.1 Air Brush and Spray Gun Options

The following table outlines the several airbrush and spray guns researched from Grex and Richpen for the painting module. Electronic references are also included for these spray guns.

<table>
<thead>
<tr>
<th>Company</th>
<th>Spray Gun</th>
<th>Electronic Link</th>
<th>Capacity</th>
<th>Action</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
E.2 Images of Various Types of Airbrushes

The following images provide visual examples of three different types of airbrushes; namely gravity-fed, siphon-fed, and side-fed. These different paint feeding mechanisms can also be applied to spray guns, and is considered in the body of this report.

Figure E.1: An example of a gravity-fed airbrush (Matco Tools 2010).

Figure E.2: An example of a siphon-fed airbrush (Grex 2010).

Figure E.3: An example of a side-fed airbrush (Grex 2010).
E.3 Parts List for Anest Iwata SGA-101 Spray Gun

The following image shows an exploded view of the Anest Iwata SGA-101 spray gun and all its components outlined in a parts list. This provides insight into the detail design and function of the spray gun for the PICARSO system.

Figure E.4: Anest Iwata SGA-101 automated spray gun parts list showing its detailed design and function.
E.4 Initial CAD Model for Spray Gun Design 1

The following image shows the initial CAD design for the Anest Iwata SGA-101 automated spray gun. This was the first attempt to model the spray gun, and was used for the proof of concept of our first detailed design of the paint module housing as outlined in Section 5.3.2 – Paint Module Design 1.

![First initial CAD design of the Anest Iwata SGA-101 automatic spray gun (Front-side).](image)

Figure E.5: First initial CAD design of the Anest Iwata SGA-101 automatic spray gun (Front-side).
F Image Processing

The following section contains content relating to the image processing development including the relevant flow charts and MATLAB code.

F.1 Flowcharts for Image Processing

The following flowcharts were developed to aid in the writing of the image processing program.
Figure F.1: Image processing flowchart – Initialisation

1. Start with first pixel

   Finished reading matrix?

   Y → 2
   N → 3

2. Is \( pv < 0.25 \)?
   Y → Set \( pv = 0 \)
   N

3. Is \( 0.25 < pv < 0.5 \)?
   Y → Set \( pv = 0.33 \)
   N

4. Is \( 0.5 < pv < 0.75 \)?
   Y → Set \( pv = 0.66 \)
   N

5. Is \( 0.75 < pv \)?
   Y → Set \( pv = 0.1 \)
   N

Figure F.2: Image processing flowchart – Greyscale filtering
Figure F.3: Image processing flowchart – Binary filtering

Figure F.4: Image processing flowchart – Output
F.2 Image Processing MATLAB Code

The following code was written in MATLAB for the purpose of the image processing of the PICARSO project. It is likely to be modified once integrated with the full scale system.

```matlab
%%School of Mechanical Engineering
%%PICARSO: Final Year Engineering Project
%%Image Processing Code
%%Ian Hooi, 1147962

clear all
cic
close all

%Run for RGB format
I=imread('uofac.jpg'); % Load the image file and store it as the variable I.
gray = rgb2gray(I); %convert to grayscale for rgb

%Run this part to read indexed image
% [I,map] = imread('mona.gif','gif'); %Read for indexed format
% gray = ind2gray(I,map); %convert to grayscale for indexed image

gray = im2double(gray); %convert to double (0->1 scale)
res_unscaled = size(gray); %Resolution of Unscaled Image
maxres = 100; %Set Maximum Desired Resolution
thresh = 0.5; %Set desired threshold value
xtrans = 0; %translation factor for keeping aspect ratio
default 0
ytrans = 0;

binary = ones(maxres); %Set up matrices as 0's initially
graynew = ones(maxres);

%Keeping Aspect Ratio
if res_unscaled(1,2) > res_unscaled(1,1) %Check if x or y res is greater
    x = maxres;
y = floor(res_unscaled(1,1)/res_unscaled(1,2)*x);
ytrans = floor((maxres-y)/2);
else
    y = maxres;
x = floor(res_unscaled(1,2)/res_unscaled(1,1)*y);
xtrans = floor((maxres-x)/2);
end

gray = imresize(gray, [y x]); %resize to x by y res
res = size(gray);

m = res(1,1);
n = res(1,2);
```
%% Standard Scaling
%gray = imresize(gray, [maxres maxres]); % resize to max res (stretch fit)

% Binary image filtering
for i = 1:m
    for j = 1:n
        if gray(i,j) < thresh % pixel value less than threshold value
            binary(i+ytrans,j+xtrans) = 0; % Set to black
        else
            binary(i+ytrans,j+xtrans) = 1; % Else set to white
        end
    end
end

% Greyscale image filtering
for i = 1:m
    for j = 1:n
        if gray(i,j) < 0.25 % Set pixel darkness depending on interval
            graynew(i+ytrans,j+xtrans) = 0;
        end
        if 0.25<gray(i,j) & gray(i,j)<0.5
            graynew(i+ytrans,j+xtrans) = 0.33;
        end
        if 0.5<gray(i,j) & gray(i,j)<0.75
            graynew(i+ytrans,j+xtrans) = 0.66;
        end
        if gray(i,j) > 0.75
            graynew(i+ytrans,j+xtrans) = 1;
        end
    end
end

% Plot Results
subplot(1,3,1)
imshow(imresize(I, [m n]))
title('Original')
subplot(1,3,2)
imshow(binary)
title(['Binary: Threshold = ' num2str(thresh)])
subplot(1,3,3)
imshow(graynew)
title('Greyscale')

% Error simulation
figure
for h = 1:m+ytrans*2
    for k = 1:n+xtrans*2
        a = rand(1,1); % Error terms
        b = rand(1,1);
        if binary(h,k) == 0
            rectangle('Position', [k-1/2+a,m-h-1/2+b+ytrans*2,1,1], 'Curvature', [1,1], 'FaceColor','k','LineStyle','none')
        end
    end
end

- 125 -
title('Error Analysis')
axis([0 n+xtrans*2 0 m+ytrans*2]);

% Gradient simulation (optimal)
figure
for h = 1:m+ytrans*2
    for k = 1:n+xtrans*2
        if binary(h,k) == 0
            for c = 0.5:-0.1:0.3
                rectangle('Position',[k-c,(m-h-c)+ytrans*2,2*c,2*c], 'Curvature',[1,1], 'FaceColor',3*[c-0.3 c-0.3 c-0.3], 'LineStyle','none')
            end
        end
    end
end

title('Optimal')
axis([0 n+xtrans*2 0 m+ytrans*2]);

% Greyscale Simulation (optimal)
figure
for h = 1:m+ytrans*2
    for k = 1:n+xtrans*2
        if graynew(h,k) < 1
            rectangle('Position',[k-1/2,m-h-1/2+ytrans*2,1,1], 'Curvature',[1,1], 'FaceColor',[graynew(h,k) graynew(h,k) graynew(h,k)], 'LineStyle','none') % positioning for rectangles
        end
    end
end
axis([0 n+xtrans*2 0 m+ytrans*2]);

%%%% Edge Detection
% edges = edge(gray);
% figure
% imshow(1-edges)
% edges_bin = edge(binary);
% figure
% imshow(1-edges_bin)
G Control Software

G.1 Control Software Flowcharts

Figure G.1: Control software flowchart – Initialisation

Figure G.2: Control software flowchart – Initialisation
Figure G.3: Control software flowchart – Manual operation subroutine.

Figure G.4: Control software flowchart – Shutdown.
H Risk Management

Risks are a large and possibly detrimental part of any design project and must be managed appropriately. Consequently, this section details the potentials risks to the success of the project, the management actions taken to minimise these risks and possible contingency plans if any risks eventuate.

Occupational Health and Safety (OH&S) issues concerning operation of the system have been approved by OH&S officer Richard Pateman. The main concern regarded where PICARSO could be operated due to paint atomisation issues.

Risks

This section lists the potential risks associated with the success of the project and the achievement of the core objectives of the project. A total of five major risks have been identified that may pose a negative consequence on the successful achievement of the project objectives.

With reference to Figure H.1 the potential project risks and their associated probability of occurrence, level of consequence and risk level are listed in Table H.1.

<table>
<thead>
<tr>
<th>Level of Consequence</th>
<th>Probability of Occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>High</td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Medium</td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Low</td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0 1 2 3 4 5 6 7 8 9 10</td>
<td>Probability of Occurrence</td>
</tr>
</tbody>
</table>

Figure H.1: Risk level chart used to determine risk factors for the PICARSO project.
Table H.1: Risk table listing the potential risks and risk levels for the PICARSO project.

<table>
<thead>
<tr>
<th>Potential Risk</th>
<th>Probability of Occurrence (0-10)</th>
<th>Level of Consequence (0-10)</th>
<th>Risk Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shortfall in project funding for relatively expensive components:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Motors &amp; motor components</td>
<td>5</td>
<td>6</td>
<td>Medium</td>
</tr>
<tr>
<td>- Spray gun &amp; spray gun accessories</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lead time delay on the ordering of motors and motor components</td>
<td>6</td>
<td>7</td>
<td>High</td>
</tr>
<tr>
<td>Lead time delay on the ordering of the spray gun and spray gun accessories</td>
<td>5</td>
<td>6</td>
<td>Medium</td>
</tr>
<tr>
<td>Lead time delay on the manufacturing of other system components</td>
<td>6</td>
<td>5</td>
<td>Medium</td>
</tr>
<tr>
<td>Communication between the personal computer and the motor controller via RS232</td>
<td>5</td>
<td>6</td>
<td>Medium</td>
</tr>
</tbody>
</table>

Risk Management

This section details the actions taken to minimise the potential risks identified above.

In order to minimise the risk posed by the chance of a shortfall in project funding, applications to both the Digital Media and Learning Competition and the 2010 University of Adelaide Open Day Innovation Fund were submitted. Additionally, several companies were approached in order to try and obtain industry sponsorship for the project. The application to the Digital Media and Learning Competition was unsuccessful, along with efforts to obtain industry sponsorship, however the authors were successful in obtaining $3000 in funding from the 2010 University of Adelaide Open Day Innovation Fund. Additionally, discounts for the purchase of motors and motor components from Maxon Motor were given, along with the kind donation of paint from Crowie’s Paints and a solenoid from APEX Automation and Robotics Pty. Ltd.

To minimise the risk of lead time delays in the ordering of motors and motor components, ordering of the spray gun and spray gun components, as well as the manufacturing of other system components two actions were implemented. The first action was to specify and order the
components, or the case of manufacturing, submit detailed designs and drawings as early as possible in the design development process. The second action was to take into account the possibility for long lead times or delays in the project schedule which, in cooperation with the first action, cumulated to the planning of the early submission of design drawings to the workshop and ordering of major system components prior to this date.

The risk of an insufficient communication speed for real-time control via RS232, between the computer and the motor controller was minimised mainly through the purchase of Maxon Motor EPOS2 motor controller boards which provided RS232, USB and analog signal control. The provision of these various communication interfaces provided alternative options for control of the motor controllers, thus minimising the potential risk of insufficient speeds.

**Contingency Plans**

This section details the relevant contingency plans that will be implemented if any of the risks identified above eventuate.

In the event that additional project funding was unable to be sourced, it was planned that the required system components could be sourced from the University of Adelaide School of Mechanical Engineering robotics and electronics resources. For example, motor amplifiers and motors would be able to be used on loan for course of the project.

If a lead time delay on the ordering of motor and spray gun related components or manufacturing of other system components eventuated, it was planned to purchase cheap and easily sourced components to be used in the PICARSO design in the meantime. A lead time delay on the ordering of motors and motor components did occur and the contingency plan was implemented which involved the purchase of a cheaper and in-stock set of motors and motor controllers. This in turn, allowed for a minimal delay in the execution of motor controller familiarisation and programming. It was also planned that, in the event of a time delay in the ordering of spray gun related components or manufacturing of other system components, the spray gun components would be borrowed from a postgraduate student in the School of Mechanical Engineering and other system components would be manufactured by the authors of this project. These system components was planned to be manufactured to provide function but not necessarily form or quality in order to minimise the time delay on other project task and deadlines while waiting for the completion of the professionally manufactured components.
In the event that communication via RS232 was not sufficiently fast for real-time control, it was planned that the USB or analog signal control functionalities of the motor controllers be used to provide a faster control interface.

Overall, the major risks to the success of the project and achievement of the core project objectives were identified and various actions employed to minimise the eventuation of these risks. However, in the event that these risks occur, contingency plans were detailed in order to avoid failure in achieving the project goals. In one case, a contingency plan was successfully implemented and the consequences that the respective risk posed were avoid, thus resulting in a minimal impact on other project tasks and overall objectives.