PICARSO
Programmable Interface Controller With Autonomous Robotic Spraying Operation

Ian Hooi, Samuel Oosterholt, Sven Paschburg, Joyce Phan, Neil Yeoh, Ben Cazzolato
School of Mechanical Engineering
The University of Adelaide
South Australia 5005
Australia
{ian.hooi, samuel.oosterholt, sven.paschburg, joyce.phan, neil.yeoh}@student.adelaide.edu.au
benjamin.cazzolato@adelaide.edu.au

Abstract

The Programmable Interface Controller with Autonomous Robotic Spraying Operation (PICARSO) system is a cable-driven painting system, with image processing capabilities. It is used as a robotic graffiti artist, with the capacity to paint both raster and vector images of any size. This paper describes the design of the system, its mechanical and electrical components, the kinematics of the cable-driven system, its imaging processing capabilities as well as results illustrating its performance.

1 Introduction

Autonomous painting systems can offer advantages over their human counterparts and may include improvements to painting speed, accuracy, reproducibility, the ability to paint in dangerous environments or without disruptions.

Several autonomous painting systems currently exist. One such system is HEKTOR, developed by Lehni and Franke [2002]. HEKTOR is a portable, two motor cable-driven painting system that is capable of painting vector images using a spray can. Another system is VIKTOR [Lehni & Rich, 2008], a four motor cable-driven robotic system, which is capable of producing vector images using chalk. These painting robots are sophisticated in their overall design and are remarkably accurate in their output.

The Programmable Interface Controller with Autonomous Robotic Spraying Operation (PICARSO) system is designed as a three cable-driven parallel planar manipulator capable of processing images. The system was scalable to allow the painting of any size image on a vertical surface. PICARSO takes an input image and paints it onto a vertical surface in one of two modes: a monochromatic raster mode (a bitmap of pixels); and, vector images using lines and curves. Using three cables allowed for PICARSO to benefit from the increased bandwidth typically afforded using four motors, as well as the lower cost associated with less actuators. PICARSO controlled the two upper motors by specifying position and the lower motor by controlling the torque.

This paper details the design of PICARSO and the results produced. In Section 2, the design of the mechanical system and the software are discussed. In Section 3, the results from tests are presented. Finally, the conclusions and future developments are discussed in Section 4.

1.1 Previous Robotic Painting Systems

In the design development of PICARSO, existing painting robots were investigated to identify and understand their successes and failures.

HEKTOR

HEKTOR was the brainchild of Jurg Lehni and Uli Franke [2002], who designed a cable-driven painting mechanism which used a dual stepper motor, cable-rigged configuration to drive a mechanically actuated spray can. HEKTOR processed a monochromatic image by determining the vector pathways which drove the spray can to the desired positions. The final HEKTOR drive system employed two stepper motors positioned in the upper left and right corners of the workspace.

Commands to the painting system were written using Scriptographer, a scripting plugin created by Lehni, to take the vector image files from Adobe Illustrator and convert them into instructions for the motors. Overall, this method allowed for sufficient control and image processing capabilities to successfully produce both raster and vector images with a high level of accuracy. An example of a painted image can be seen in Figure 1, which gives evidence of Hektor's painting ability.

Figure 1: An output produced by the HEKTOR painting system. The spray can end-effector can clearly be seen. [Lehni & Franke, 2002]

VIKTOR

VIKTOR was a vertically mounted image producing cable-driven manipulator designed by Jurg Lehni and Alex Rich [2008]. This system, an evolution of HEKTOR, used a four motor cable-driven design to guide its end-effector in the vertical plane. Additional motors were utilised in order to increase the bandwidth, stability and manoeuvrability of the end-effector across the canvas in comparison to Hektor's dual motor design. Additionally, servo drives were implemented instead of stepper motors, as the increase in complexity of the end-effector and cable rigging system required more complex and flexible control strategies.

The wall mounting system comprised of modular units each housing a single motor, motor controller and spooling system. This allowed for portability and scalability of the 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.

Figure 2: A chalk drawing produced by the VIKTOR painting system. The end-effector being manipulated by a four motor configuration can clearly be seen. [Lehni & Rich, 2008]

2 System Overview

The objective of PICARSO was to reproduce an input image onto a vertical canvas using a spray gun housed in the end-effector. The system was designed to be portable and scalable so that it could be attached and operated on any flat vertical surface. PICARSO was developed to output a binary (monochromatic) images painted by either raster (pixel-by-pixel) or vector (continuous line segments) modes.

A three-motor X-configuration was selected to manipulate the end-effector, as shown in Figure 3. This allowed for fast motion of the end-effector as the downward tensioning force was not restricted by gravity as with the two-motor HEKTOR. The X-configuration was accomplished by feeding the lower cable horizontally to a pulley fixed at the lower right corner of the reference frame. This pulley guided the cable to the bottom of the end-effector, which was then fixed to a point at the lower left corner of the reference frame.

The lower cable's pulley was attached via a sling around the body of the end-effector. This allowed the pulley to move freely underneath the end-effector, which minimised the tendency for the lower cable to tilt the end-effector during operation. Tilting (rotation of the end-effector so it is no longer normal to the workspace) occurred when the end-effector was positioned at the extreme left or right hand side of the workspace. Tilting of the end-effector was undesirable as it would result in positional errors of the tool centre point, which would cause inaccuracies when painting. However, the employed X-configuration, along with the lower cable tensioned by the lower motor in current mode, provided effective stabilisation of the end-effector and minimal tilting, which allowed for accurate positioning.

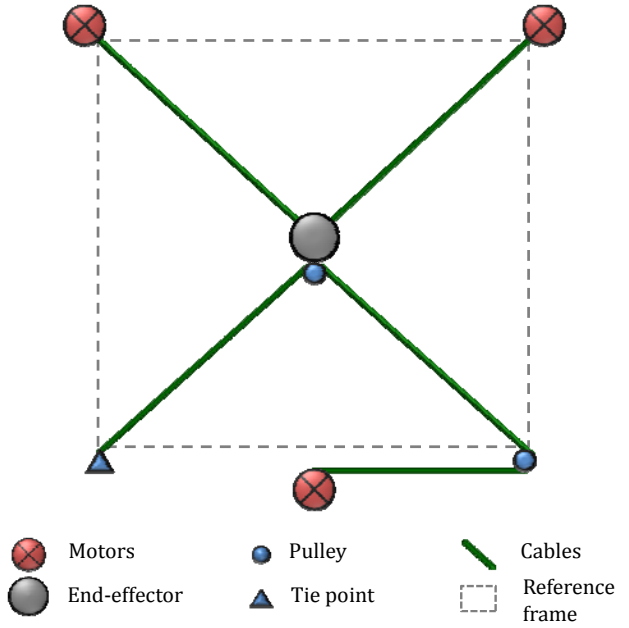


Figure 3: Schematic of the components contributing to the X-configuration employed in the final design of PICARSO.

PICARSO transforms an input image using the developed image processing software to produce a desired output. The image processing output was converted into commands by the control software. These commands were then sent to the electronics, which manipulated the end-effector and painted the image. Detailed descriptions of these sub-systems of PICARSO are provided in the following sections.

2.1 Mechanical System

Within PICARSO's mechanical system, motor mounts were designed which provided the mounting structure for a motor, a motor controller, as well as the cable system. Furthermore, the end-effector housed the painting mechanism. The following sections outline the design development of the components of the mechanical system.

Motor Mounts and Cable System

The motor mounts were designed to provide a modular and portable structure to attach the drive system, control system and cable system to the vertical surface. The drive and control systems consisted of the motor assembly and control hardware respectively, while the cable system comprised of the cables, the spool and the cable feeding system. The final design of the upper left motor mount is seen in Figure 4.

Each motor was mounted parallel to the painting

surface on an aluminium angle bracket fixed to a base plate designed to mount directly to a vertical painting surface with bolts.

To control the motion of the end-effector, a double spool and coupled bearing were attached to each motor to wind two cables simultaneously. Two cables per motor were used to keep the end-effector normal to the workspace and reduce the kick-back when air was expelled from the spray gun. Also, in the event of one cable failing due to excess tension, the second cable would be able support the load, minimising damage to the system and increasing safety.

The cables are reorientated from the winding plane to the working plane through eyelets and pulleys attached to the motor mounts. The lower plate uses fairleads in place of the pulleys to allow for 180° sweep of the cable.

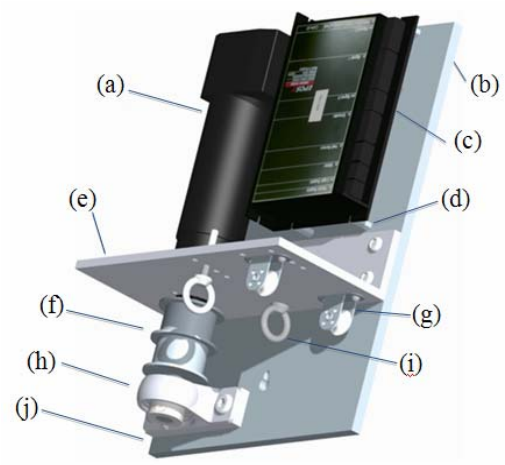


Figure 4: PICARSO's motor design showing: (a) Maxon EC45 motor, (b) base plate, (c) Maxon EPOS2 motor controller, (d) hex spacer, (e) angle bracket, (f) double spool, (g) pulley, (h) bearing assembly, (i) eye-bolt and (j) bearing spacer block.

End-Effector

The aim of the end-effector design was to secure the painting mechanism for the PICARSO system and for it to be integrated into the mechanical system for positioning and stabilisation of the painting mechanism. The painting mechanism chosen for the PICARSO system was an automatic pressure-fed spray gun (Anest Iwata SGA-101).

Automated spray guns are robust and provide repeatable results through a solenoid controlled spray. Analog signals from a data acquisition (DAQ) card were used to switch an electronically actuated solenoid to control the spray gun's operation.

The paint used for pressure-fed spray guns is located externally from the spray gun in a pressurised paint canister. Thus, when the paint is used up during operation, the weight of the spray gun remains unchanged, and hence does not affect the dynamics of the system, thereby making control easier. The chosen pressure-fed spray gun has the ability to paint onto vertical surfaces and to house enough paint to paint a 3×3 metre surface area in one uninterrupted operation.

The assembled end-effector housing the spray gun is shown in Figure 5. This design comprised of an aluminium sleeve which encased the spray gun from behind. This spray gun and sleeve were located inside a larger cylindrical sleeve which allowed free rotation of the spray gun about the axis of its nozzle tip. This ensured that as the end-effector moved through the workspace, the positional error of the nozzle tip would be minimised, despite rotation of the outer cylindrical sleeve. The outer sleeve was also designed such that it did not obstruct the air and fluid lines of the spray gun in the smaller inner sleeve. Three pairs of 120° spaced eyelets were placed around the outside of the larger cylindrical sleeve which allowed connection to the motor mounts through the cables. An exploded view of the housing can be seen in Figure 6.

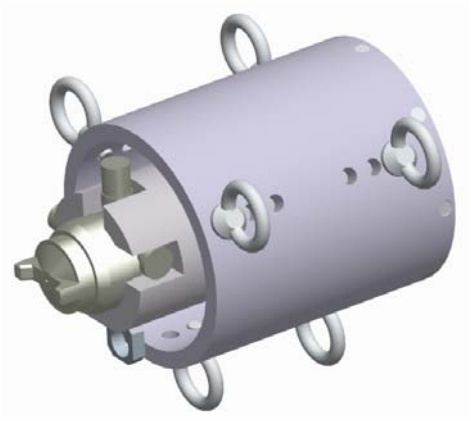


Figure 5: An assembled view of the end-effector design with the chosen spray gun housed within.

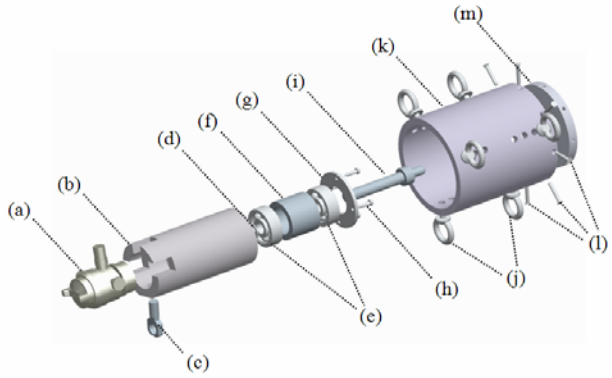


Figure 6: An exploded view of the SGA-101 Spray Gun with design housing including: (a) SGA-101 spray gun, (b) inner aluminium sleeve, (c) fitting knob, (d) circlip, (e) bearings, (f) bearing spacer, (g) inner sleeve cap, (h), button screws, (i) sleeve shaft, (j) eyebolts, (k) outer aluminium sleeve, (l) 8 × bolts, and (m) outer sleeve cap.

2.2 Electronic System

The electronics provided communication between the software and hardware to control the system. The electronics consisted of the drive system, control system and complementary hardware.

The drive system encompassed the motors and any additional hardware required to manipulate the end-effector by spooling cables to their desired lengths. Three Maxon Motor EC45 motors with a 26:1 gearhead were selected, which were capable of a maximum continuous

torque of 6.6Nm, and operating at approximately 175rpm.

The control system consisted of the hardware used to send command signals to the system actuators, which consisted of three Maxon Motor EPOS2 70/10 motor controllers, and three optical encoders and gear head assemblies. The additional hardware implemented in PICARSO included a National Instruments PCI-6221 DAQ card to convert the digital signal motor command to an analog signal that could be sent to the motor controller as an analog input. Control using analog inputs was selected to command the position of the two upper motors in order to eliminate latencies associated with the motor controller firmware when using serial or USB communication.

As previously stated, the spray gun used an electronically controlled solenoid valve to actuate when required. A three-way solenoid valve was used as a switch for the air inlet to the spray gun and controlled using an analogue output from the DAQ card. Through testing, this was deemed to be appropriate for the switching frequencies required for this application.

2.3 Kinematics

The control software required inverse kinematics to convert the image processing output into motor position commands used by the upper two motors to move the end-effector. To determine the inverse kinematics, the workspace, reference frames and relative positioning of the components of the system are defined, as seen in Figure 7. Due to being controlled by specifying torque, inverse kinematics for the bottom motor was not required.

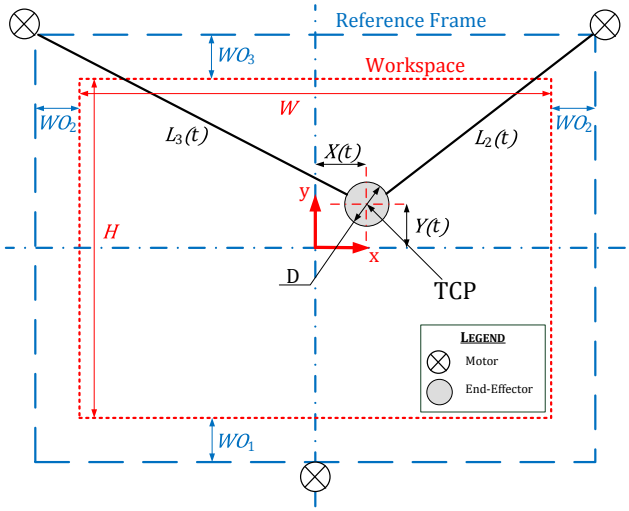


Figure 7: A diagram which defines the workspace and reference frame for PICARSO.

The inverse kinematics of the system were derived as follows:

$$L_1(t) = \left[X(t)^2 + \left(\frac{H}{2} + WO_1 + Y(t) - \frac{D}{2} \right)^2 \right]^{\frac{1}{2}} \quad (1)$$

$$L_2(t) = \left[\left(\frac{W}{2} + WO_2 - X(t) - \frac{D\sqrt{3}}{4} \right)^2 + \left(\frac{H}{2} + WO_3 - Y(t) - \frac{D}{4} \right)^2 \right]^{\frac{1}{2}} \quad (2)$$

where:

- $L_1(t)$ – Length of Cable 1
- $L_2(t)$ – Length of Cable 2
- $X(t)$ – x-coordinate of end-effector position
- $Y(t)$ – y-coordinate of end-effector position
- W – Width of the workspace
- H – Height of the workspace
- WO_1 – Vertical offset of Motor 1 from the workspace
- WO_2 – Horizontal offset of Motor 2 & 3 from the workspace
- WO_3 – Vertical offset of Motor 2 & 3 from the workspace
- D – Diameter of the end-effector

For a cable-driven parallel planar manipulator, each Cartesian coordinate position represented a unique set of cable lengths as well as a unique set of motor positions. Therefore, there were no issues associated with non-unique inverse kinematics solutions occasionally experienced by other types of manipulators [Merlet & Gosselin, 2008].

2.4 Software System

PICARSO'S software comprised of image processing and control software, as well as a graphical user interface (GUI) to control and add interactivity to the system. The software was programmed using MathWorks' MATLAB in order to utilise its Image Processing, Data Acquisition and GUI development toolboxes.

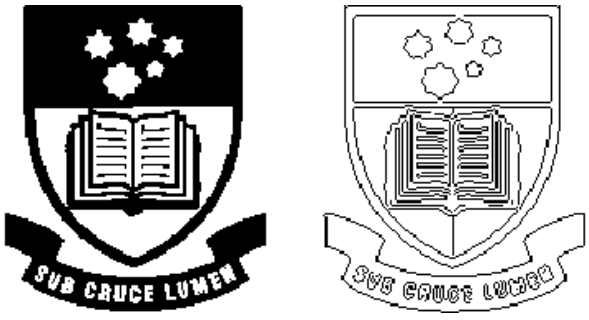
Image Processing

The image processing software was required to take any standard input image and convert it to a form which was suitable to be output to the control software. Firstly, the input image was read in a colour raster form as a matrix, with each entry corresponding to a pixel of the image. The image then underwent several transformations using MATLAB'S Image Processing Toolbox's inbuilt functions. A user specified value for the maximum resolution determined how detailed the painted image would be. Additionally, the image was either directly resized to a user specified desired resolution or the aspect ratio was taken into account while resizing (to account for available workspace). The image was then transformed into a greyscale image, with its entries being normalised to double precision. This process scaled the matrix elements between 0 (black) and 1 (white), depending on the intensity of the pixels.

In order to construct the binary image, a threshold filter was used to filter pixels that were less than a user specified threshold value. As all pixels in the image matrix had been normalised from 0 to 1, the threshold value was also set to be between this interval. Therefore, the filtering algorithm set a pixel to 0 if its magnitude was below the threshold value and set it to 1 otherwise.

The image could be converted to both a filled and edge binary format as shown in Figure 8. A filled binary image was a direct conversion from greyscale to binary based on the threshold setting, whereas an edged binary image extracted the outlines from the input image and ignored the coloured in areas using an inbuilt MATLAB function. Edge images were advantageous

because they required less black pixels to be painted, which allowed for faster painting time, when compared with a filled image, at the expense of image detail.



(a) (b)

Figure 8: A MATLAB output of the two types of binary images showing (a) filled and (b) edge images.

The final step was to output the binary image to the control software. Two output types were considered for the image processing software: raster and vector modes.

Using raster mode, the resulting binary image could be output as a matrix to the control software. In this case, the image was painted one pixel at a time, which was shown through simulation and testing to be slow, especially for high resolution images.

The software plotted a theoretical raster output on the computer screen, which depicted the expected output of PICARSO once painted. This image accounted for the finite pixel (paint spot) size as well as inter-pixel spacing. An example of the conversion from the original image, Figure 9, can be seen in Figure 10.

Figure 9: The original input image to be processed by the image processing software. The image is a (raster) JPEG format. [Knoth, 2010]

Figure 10: The theoretical (expected) raster output from the image processing software of the original image shown in Figure 9.

Vector mode, on the other hand, connected adjacent black pixels into chains or vectors in order to accelerate the painting process. An algorithm was developed based on the PORTRAYER [Benedettelli, 2008] and Eric's X-Y plotter [2007], which converted the binary raster image into a series of vectors. The application of which, when applied to the image in Figure 9, is shown in Figures 11 and 12. When using this output method, multiple pixels could be painted at once, thus, increasing the painting speed.

Several search masks were developed, which modified how an image was mapped. These masks modified the search direction preferences of the algorithm which affected the number of line segments required to be painted, which in turn affected the total painting speed of an image. However, the optimal search mask varied for each input image and therefore this setting could be changed as necessary.

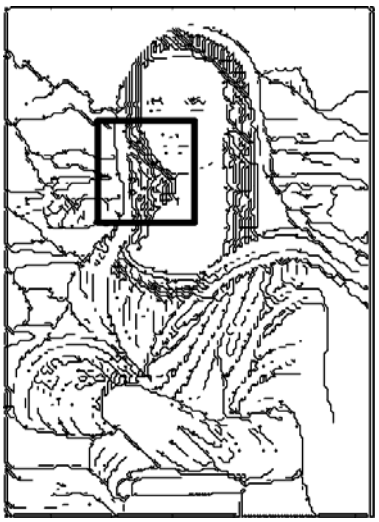


Figure 11: The theoretical vector output of the Mona Lisa in Figure 9. The black box shows the zoomed area in Figure 12.

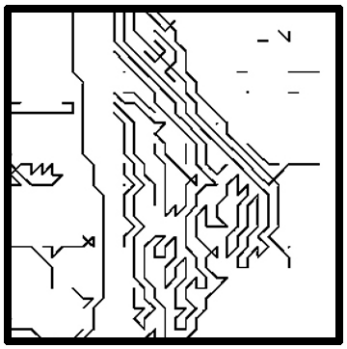


Figure 12: A zoomed in view of the vector output seen of the image in Figure 11, showing how the algorithm converts the image into a series of lines.

Control Software

The control software was developed in order to control the hardware of the system and consisted of two control modes, inverse kinematics and output commands shown in Figure 13. Additionally, the control software incorporated a GUI, which was implemented to provide a user interface to the PICARSO system that integrated the image processing and control software components into a single user environment. The GUI provided options to the core functionalities of the system including the two control modes (raster and vector mode). These control modes provided the core painting functionality of the system. Two additional modes were developed for the control software including a manual end-effector positioning mode and a dedicated spray gun control mode, which were used for testing and maintenance purposes.

The high level flow of the control software shown in Figure 13 displays the process of converting an input image output to Cartesian coordinates and then applying inverse kinematics to position the end-effector. Upon starting of the control software the user is presented with a GUI incorporating all the relevant image processing and system operation settings. The image processing output defined by the image settings is taken and passed into either raster or vector mode algorithms. These algorithms perform the conversion of the image output to Cartesian coordinates. After the image output is passed through the appropriate algorithm, the Cartesian coordinates are in joint space and buffered for the painting operation. Once buffered, the painting operation begins and motor commands are retrieved and sent via the output commands component to drive the motors and actuate the spray gun. Upon completion of the painting operation, the flow returns to the GUI, ready for the next instruction.

The output commands aspect of PICARSO's software involved the direct control of all of the system's outputs without any high level "intelligent" control. Specifically, the output commands involved driving (via the DAQ) the motors to allow the end-effector to move, the actuation of the solenoid to activate the spray gun at the appropriate time and position, as well as initialisation functions required for the automation of the system.

To communicate with the Maxon EPOS2 motor controllers, the manufacturer specified serial RS232 communication protocol was used to transmit and receive data over the EPOS2's RS232 serial port. Using this communication structure, the master computer was able to send instructions to command each of the three EPOS2 motor controller slaves.

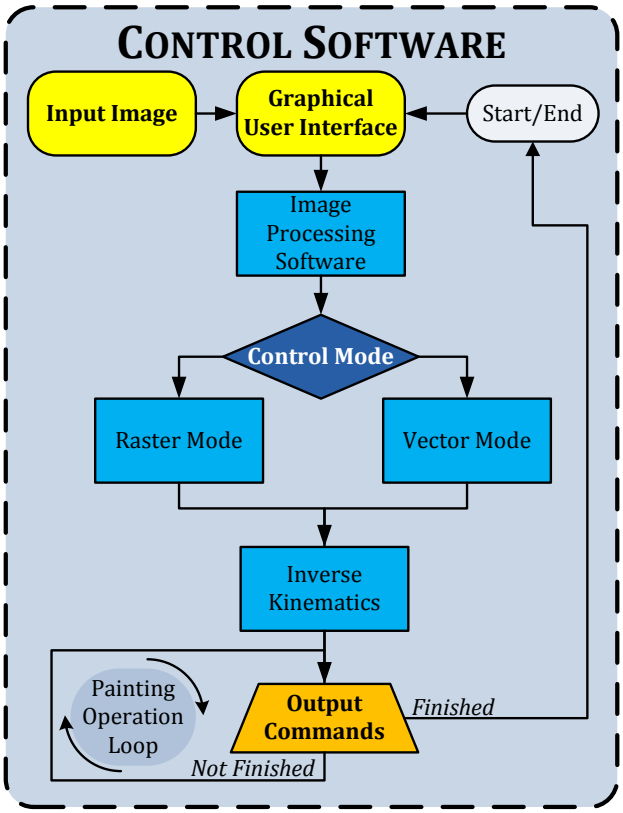
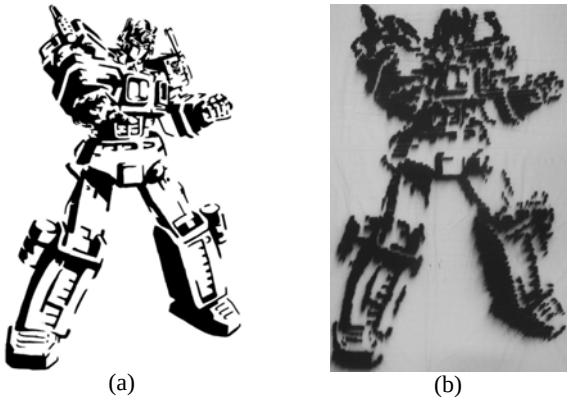


Figure 13: The structure of the high level control software.

Additionally, in order to stabilise the end-effector for improved position control and accuracy of the system, a novel drive system was implemented. The control software drove the upper two motors by specifying position and the lower motor by specifying a current. Specifying a desired current enabled the lower motor and corresponding cable to provide a tensioning force, or torque, and as a result acted as a stabiliser for the end-effector, much like a spring. This force aided in keeping tension in the upper cables, as well as damping any oscillations of the end-effector during system operation.

3 Results and Illustrations

PICARSO was implemented to produce raster and vector images on 2.4×2.4 metre vertical sheets. The precision and accuracy of the image produced, as well as the time taken to paint the image, were among the numerous metrics derived to assess the performance of the system for commercial viability, which were observed during the operation of PICARSO or from the image produced. Figure 14 is an example of the raster fill painting produced by PICARSO.

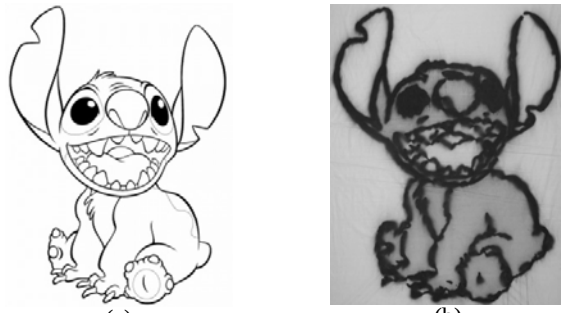


(a) (b)

Figure 14: A raster fill painting of a Transformer [Picasa Web, 2008] showing, (a) the original image, and (b) the painted image.

A spray duration of 0.20 seconds produced pixels of consistent intensity and diameter of 11±1 mm. Accuracy of each pixel from one end of the workspace to the other varied by ±5 mm, while the repeatability of pixel spraying varied by ±2.5 mm due to environmental factors, such as wind. Wind vibrated the end-effector, varying its distance from the canvas by ±25mm and caused diffusion of the paint: tight pixels when close to the canvas, and larger blury pixels when far from the canvas. The production of the Transformer image took over three hours due to the stop-start spray and movement method, which was improved by the vector images.

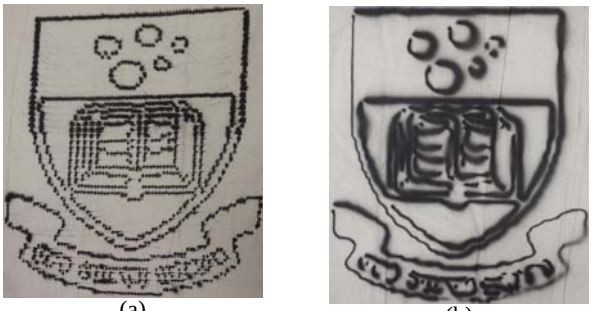
Figure 15 is an example of an image produced by PICARSO's vector fill functionality. In comparison to the raster image, this vector image was completed in twenty minutes with end-effector velocity of 100mms^{-1}, and produced a line of width 11±2 mm.



(a) (b)

Figure 15: A vector fill painting of Stitch [Supercoloring, 2008] showing, (a) the original image, and (b) the painted image

To directly compare the raster and vector outputs, two 1×1 metre images of the University of Adelaide logo were produced using both functionalities. The results can be seen in Figure 16. The time taken to produce the raster image was 41 minutes, as opposed to 7 minutes for the vector image. To produce an image using vector mode provided a dramatic reduction in painting time when compared to the image produced in raster mode.



(a) (b)

Figure 16: A comparison between a (a) raster edge image and (b) vector edge image. Note the top horizontal line in (a) was painted horizontally but has been distorted when the photograph was taken.

A trade-off for the high image production speed was that the line segments produced by vector were less accurate than those produced by raster. Rectangular corners were observed to be rounded by up to 20 mm radius and lines greater than 80 mm in length did not join together at completion by up to 15 mm. This was minimised by reducing the speed of the end-effector, but resulted in line thickness greater than 11 mm which was undesirable for image quality.

A summary of the performance and capabilities of PICARSO are presented in Table 1.

Table 1: A summary table of results from all tests.

Test Metric	Result
Accuracy (mm)	± 5
Repeatability (mm)	± 2.5
Workspace Resolution (mm)	10 to 100
End-Effector Velocity (mms^{-1})	Optimal – 100 Minimum – 1 Maximum – 200
End-Effector Acceleration (mms^{-2})	Optimal – 100 Minimum – 1 Maximum – 200
Raster – Edge Image	2487s = 41.45 mins
Vector – Edge Image	442s = 7.37 mins
Workspace Area (m)	Min: 0.0 × 0.0 Max: 2.4 × 2.4
Workspace Scalability (m)	Min: 0.2 × 0.2 Max: 8.4 × 8.4
Spray Duration (s)	Optimal – 0.20 Minimum – 0.10 Maximum – 1.00
Pixel Size (mm)	Optimal – 10 to12 Minimum – 5 to 6 Maximum – 24 to 27

Accuracy and precision tests revealed that the system was able to position with high accuracy and precision relative to the available workspace area of 2.4 ×

2.4m. Additionally, the optimal workspace resolution for raster painting was determined to be 10mm. Overall, accuracy and precision tests showed the system was capable of accurately manipulating the end-effector via the hardware, while inverse kinematics computed within the control software were accurate in commanding the hardware of the system.

Tests revealed that the optimal velocity and acceleration (detailed in Table 1) produced the best image results. Additionally, painting time tests revealed that producing vector paintings markedly reduced the time required to paint a given image when compared to painting a raster image. Tests were also performed to determine the scalability of the system in terms of the workspace area and reference frame. Results showed that the workspace area was scalable up to 8.4 × 8.4m.

Raster painting tests identified that the optimal pixel size for raster painting, which were found to be 0.20s and 10 - 12mm, respectively. Also, results showed that the system was able to accurately reproduce images with consistent proportion and shape to the original image. This was verified for simple and arbitrary shapes, and most importantly for more complex images. Minor discrepancies and inconsistencies, however, did occur due to environmental conditions, stability of the end-effector and the rigidity of the painting canvas during operation. However, these errors did not impact greatly on the quality of painted images.

4 Conclusions and Future Work

The aim of PICARSO was to design and build a robotic system capable of painting an image onto a vertical canvas. The system was to be portable, scalable and able to autonomously process an image into instructions for reproduction. The mechanical system has been designed and built to meet these design specifications, and the software has been developed to achieve the accurate painting of large images in raster and vector modes.

Applications in marketing and advertising are being explored as PICARSO is being tested to paint on external walls of buildings. Further improvements being considered for the system include integration of hardware to achieve colour images, upgrading the spray gun to achieve a higher resolution, and wireless communication and batteries to further improve the portability of the system.

Acknowledgements

The authors would like to acknowledge the support of The University of Adelaide Open Day Innovation Fund, Maxon Motors Australia, Anest Iwata Australia, Apex Automation, Crowie's Paints and Dr Peter Kalt.

References

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

[Convict Episcopal de Luxembourg, 2007] Convict Episcopal de Luxembourg, 2007, Erik's XY-Plotter, viewed 11th April 2010, <http://www.convict.lu/Jeunes/ultimate_stuff/Erik_s_x_plotter/E_xy_plotter.htm>.

[Knoth, 2010] Knoth, C. 2010, C Knotes, viewed 10 September 2010, <http://catinkacards.tripod.com/pictures/mona_lisa_bw_2.JPG>

[Lehni & Franke, 2002] Lehni, J. & Franke, U. 2002, Hektor, viewed 29 December 2009, <http://www.hektor.ch/Book/Hektor.pdf/>.

[Lehni & Rich, 2008] Lehni, J. & Rich A., 2008, *A Recent History of Writing & Drawing*, Institute of Contemporary Arts Publishing, London.

[Merlet & Gosselin, 2008] Merlet, J. & Gosselin, C., 2008, 'Parallel Mechanisms and Robots', Springer Handbook of Robotics, Part B, Ch. 12, pp. 269-285.

[Picasa Web, 2008] Picasa Web, 2008, *Transformers Stencil*, view on 15th October 2010, <http://picasaweb.google.com/lh/photo/ICcqfXh4HVDrMZDs7JlZnA>.

[RoboPainter Texas, 2008] RoboPainter Texas 2008, RoboPainter Texas, Texas, viewed 3 January 2010, <http://www.robopaintertexas.com/>.

[Supercoloring, 2008] Supercoloring, 2008, *Stitch Stencil*, viewed 22nd October 2010, <http://www.supercoloring.com/wp-content/thumbnail/2008_12/stitch-coloring-page.gif>.