Development of a Marker Following System for use in the Mariokart System

Zachary Taylor
zjt14@uclive.ac.nz

Coauthors: Simon Richards, Wim Looman and Henry Jenkins
scr52@uclive.ac.nz wgl18@uclive.ac.nz hvj10@uclive.ac.nz

Supervisor: Dr. Andrew Bainbridge-Smith
andrew.bainbridge-smith@canterbury.ac.nz

Department of Electrical and Computer Engineering
University of Canterbury
Christchurch, New Zealand

***Abstract*—An autonomous go-kart system, nicknamed Mariokart was being developed. It required a simple algorithm to demonstrate its abilities and test its systems. To meet this need an algorithm was developed to follow the path of a marker. Three computer vision techniques color detection, SURF and chessboard detection were evaluated to see if they could be used to locate the marker. SURF performed slowly and only at very short ranges, colour detection was found to be too sensitive to lighting changes but otherwise a viable option. Chessboard detection was found to meet all the needs of the system with the exception of having a reasonably short range. Once the method of marker detection was in place a method for allowing the kart to follow the same route as the marker was implemented. This utilized the karts on board sensors to estimate its location. The system was tested but never mounted to the Mariokart due to delays in the development of systems it required to interface with the kart.**

I. Introduction

An electric go-kart was to be converted to allow it to navigate autonomously. To this end the system, nicknamed Mariokart was being converted so that it could be controlled from a laptop. This control system required a program to be written which would give the kart some basic autonomous functionality. This program was to function as a proof of concept of the karts abilities as well as a method to test the systems that had been added to the kart. It had been decided that this program should allow the kart to follow a marker.

The system that was being designed allowed the systems on the kart to interface through a laptop to 5 boards located on different areas of the kart. These boards handled all the low level control of the kart providing the laptop with functions to set the steering, brake and accelerator position. It also provided functions to access sensors located on the kart through fairly high level commands. The only sensor present on the kart at the start of the development was a speed sensor however a large number of I/O ports of different types allowed for the addition of extra sensors.

The developed program was to run on the on-board laptop. It had to operate in real time recognizing the marker, finding its location and outputting the path to travel on with minimal lag. The kart was expected to be able to operate in both indoor and outdoor environments in a large variety of conditions without losing the marker. The system and any additional equipment and sensors required had to be low budget as the total budget for the entire conversion was less than $1500.

II. Marker Location Methods

A large variety of location methods were looked at when determining what type of sensors should be purchased for the kart system. Most modern autonomous vehicles rely heavily on scanning laser sensors as their main method of viewing the surroundings. Scanning lasers as their name implies rapidly move a measuring laser over the environment forming a matrix of distances to provide a 3d view of the world [1]. While these lasers are extremely useful they were discounted as a possibility early due to their high price tag with even the most basic models starting at $2000. The Microsoft Kinect camera was briefly looked at as an alternative solution that would provide 3d information about the world [2]. This camera uses structured light combined with an IR (inferred) camera to detect the location of objects in its field of view. This system was also quickly discounted however as it had a maximum range of only 3.5m which was too limiting for the application of tracking a marker from on board a kart. A second aspect that counted against the Kinect was that the IR pattern used by it was washed out by direct sunlight making outdoor use impractical.

GPS systems, while briefly examined as a possibility were not given any serious consideration due to their inability to work reliably in an indoor environment. One solution that was given serious consideration was using a marker that output a RF (radio frequency) signal and having three RF receivers on the kart use the relative strength of the signal to triangulate the position of the marker. After further research it was found however that distance estimation using the level of RF detectors that were within the budget was only accurate to around 1m under ideal conditions [3]. This limitation meant that the receivers would have to be placed far apart to minimize the effect the error would have on the direction the triangulation would calculate the signal to have come from. On a mobile kart system where the close proximity of the sensors was unavoidable this limitation made the method infeasible.

The sensor system that was decided upon to develop was a computer vision system. This system utilized a web-cam mounted to the front of the kart. A unique marker was to be used that could be located in the image. This method had several advantages over the other methods considered. It was a cheap and easy to mount solution with only a web-cam required that would attach straight to the laptop with a USB cable. It provided a large degree of flexibility with a large suite of different algorithms and methods able to be implemented and tested for no modifications to the hardware the system was using. Finally it was the only system that showed promise in being able to locate a marker accurately at distance and within the budget. It was for these reasons that the computer vision system was the only option pursued for development.

III. Computer Vision Theory and Background

A large number of algorithms exist for locating a marker in a video feed, however to be appropriate for the task at hand the algorithms had to have several key features besides simply finding the marker.

- The algorithm had to be able to reliably give the position of the marker relative to the kart including an accurate estimate of the distance from the kart
- It had to work even when the marker was moving quickly across the camera and at a large distance from the camera.
- It had to be robust to the large variations in lighting conditions and backgrounds that it would experience when mounted to the kart.
- Finally it had to be able to be performed in real time giving out information on the position of the objects location with little lag and at a high refresh rate.

Based on these criteria three methods were found that showed promise at being able to meet these requirements.

Colour detection was the first process looked at. This approach takes the cameras image and converts it to HSV (hue, saturation and value) space [4]. In this space each pixel is classified by its hue or colour, its saturation which is the intensity of the colour and its value which represents how light or dark the colour is. This hue is assumed to vary very little for the marker. For the most simple form of colour tracking the image is tresholded for a range around the colour of the marker being tracked. This process leaves only the desired colour present in the resulting image. The x and y positions are found by finding the location of the centre of the resulting blob in the thresholded image. The distance of the marker is taken to be proportional to the inverse of the square root of the markers size in pixels.

Chessboard detection is often used in augmented reality applications to get the position of a marker relative to a camera. It is used so that a 3d model can be displayed with the correct size and orientation as if it was actually present in the scene [5]. This algorithm uses a chessboard of known size as its marker. It thresholds the image and applies a Harris corner detector to find the corners in the image. This information is combined with a Hough transform to find the lines on the chessboard and once all the corners and lines have been found the image is searched for a pattern of them that would match the chessboards geometry [6]. This method considers all of the intrinsic parameters of the camera as well as the extrinsic transformation for the chessboards location in the scene. This means that the position of the chessboard marker can be accurately computed even taking into account the curving effect of the cameras lens at the corners allowing for more accurate marker location.

The third method examined for detecting a marker in a scene was the SURF (speeded up robust features) algorithm. This algorithm operates along very similar lines to the SIFT (scale invariant features transformation) algorithm except operating at a greater rate, hence the reason for the ”speeded up” in its name. The SURF algorithm matches points that are common to two separate images. In this case the images are a unique marker and an image of a scene in which the marker is located. The algorithm operates by first selecting a large number of unique points in the first image. Finding these points involves finding areas of the image that are unique in their surroundings and that can be easily identified regardless of changes to the viewpoint [7]. By themselves these points could not be located in a new image as a one dimensional point cannot be identified as different from any other point and so a small window surrounding each point is also saved. Each new frame of the scene is then scanned for areas that match these windows using Harr like features to make this match [8].

Once the points are matched another algorithm RANSAC (RANdom SAmple Consensus) was used to extract the size and orientation of the marker from the scene. RANSAC itself is not a computer vision algorithm but an iterative method to find parameters of a mathematical model within noisy data where redundancy in the number of points exists [9]. RANSAC works by randomly selecting four points in the image and from this working out the transformations the picture has undergone to fit these four points in the scene. Using this transformation a calculated location is found for all the other points. The distance between where a point is calculated to have been and where it actually lies in the image is found. This gives an estimation of the error in the transformations that the picture has undergone. If this error is small the transformation is kept and taken as correct, if it is large the process is repeated with four new points [10].

IV. Implementation of Solutions

Colour detection, SURF and chessboard detection were all implemented to evaluate their performance for tracking a marker in the required environment. To aid in this implementation extensive use was made of the Opencv computer vision libraries [11]. These libraries were used as they contained implementations of a large number of computer vision algorithms that had been thoroughly tested by a large community and optimized to run as effectively and efficiently as possible.

The first algorithm implemented was the simple colour based approach. In this approach a bright blue ball was used as the marker to track. Initially the scene was thresholded for a value that was approximately a bright blue and opening (that is erosion followed by dilation) was used to fill any holes in the image of the ball and remove background noise. The output of this process can be seen in Figure 1. A rough estimation of the balls location was then made using its x and y position in pixels from the centre of the camera. The z location of the ball was taken to be proportional to the inverse of the square root of the size of the ball in pixels. These measures of x, y and z were extremely crude however they provided adequate for the testing that was performed on the colour method.

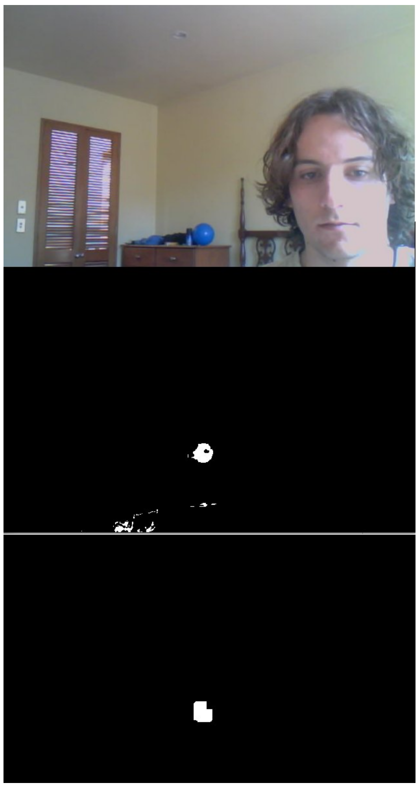
Fig. 1. Detection of marker ball using colour

For both the SURF and chessboard algorithms the intrinsic parameters of the camera had to be found so that the size of an object on the screen could be translated from the arbitrary pixel co-ordinate system used by the images to real world distances. These parameters were found by using the camera to take a series of photos of the chessboard from different angles and distances and using the geometry of the chessboard to extract the cameras parameters [12]. Once this had been performed the parameters were saved to a file that was used by both algorithms. This calibration was a small disadvantage of these algorithms however as it only had to be performed once for a web-cam it did not diminish the practicality of their use by much.

The SURF algorithms marker was a printout of a cut away of an aeroplane. This image was used as it contained a large amount of strong lines and detail that meant that a large number of unique and easily identifiable points could be located on the image. An example of the SURF algorithm working can be seen in Figure 2.

Two chessboards were constructed and tested with the chessboard registration algorithm. A large 6 by 5 chessboard with 100 mm squares and a smaller 7 by 8 chessboard. The larger chessboard was to be used with the kart once it was running. The smaller one was used for calibrating the camera and the testing covered in this report. The smaller board can be seen being used in Figure 3.

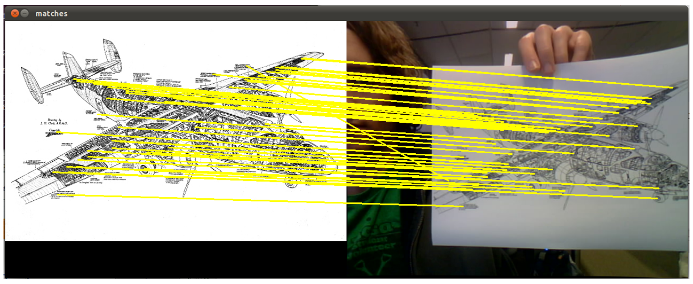

Fig. 2. The SURF algorithm matching points in the marker image to the scene

V. Performance of Solutions

The algorithms were put through a series of tests to guage how well they would perform in the system. All the tests were performed on a 2.1 GHz notebook with an Intel Centrino Duo processor. The webcam was a Creative Labs Laptop Integrated Webcam. This webcam had a resolution of 320 by 240 pixels and operated at 20 fps.

A. *Robustness to lighting*

The robustness to light was tested by setting up an algorithm to recognise its marker in a room with a large amount of natural light. The lighting was then changed by closing the curtains and seeing if the object was still detected, once this was done the rooms lights were turned on. Colour detection performed the worst in this test requiring recalibration before it could detect the target after every change in lighting. Its robustness could be improved but this caused issues with background noise, this is gone into in detail in subsection E. The issue with lighting arose because while the hue of the marker being located remained constant changing the lighting levels altered the hue the camera perceived the marker to be by a small amount. This problem stemmed right from the fundamental method by which the camera detects light and so very little could be done without recalibrating to attempt to account for this change [13]. The SURF algorithm handled almost all lighting changes without issue. The one situation in which it failed was if enough glare was coming off the marker it was locating to completely obscurer most of the features it was trying to find. The chessboard algorithm worked under every tested lighting condition.

B. Frame processing speed

The FPS (frames per second) for each system was recorded under normal operation. The colour detection algorithm required easily the least processing power of all the computer vision methods looked at running at a solid 20 fps only limited by the speed of the camera it was receiving the images from. The SURF algorithm operated very slowly originally operating at 0.8 FPS on locating a 640 by 480 pixel reference image in the scene. This speed was increase to 2 FPS by decreasing the reference image to 320 by 240. This was the lowest quality reference image which could be reliably placed in the scene. Chessboard detection operated slowly if the chessboard had not been located in the previous frame only operating at a rate as low as 2 fps. Once the chessboard was found however the algorithm used the previous location to find it more quickly raising the frame rate to around 15 fps.

C. Robustness to motion

Robustness to motion was tested in a fairly subjective manner. The marker was held roughly 1m from the camera and shaken until detection was lost. With colour detection regardless of how fast the marker was moved detection was never lost, though the detected size decreased by up to around 50% as the image of the marker became a blur. With SURF detection only slow and careful movements would prevent the camera losing the target. Chessboard detection would lose the board if any quick or jerky movements were made. While this test was quite crude it clearly showed that SURF would be too susceptible to motion blur to be appropriate. It also showed that chessboard detection was weak to it and a higher quality camera or some pre-processing might be needed to help reduce the effect of motion blur when the system is attached to the kart.

D. Maximum distance for detection

The maximum distance the algorithm could detect a marker from was tested next. To make this a fair test for all systems the marker size for each system was set to an A4 page with the colour detections blue ball replaced by a sheet of blue card. The test was performed by walking away from the camera holding the marker and recording the distance at which the target was lost for the first time. The process was repeated three times and the results shown in Table I. SURF performed easily the poorest of the solutions with the marker needing to take up a large portion of the field of view of the camera before recognition occurred. The chessboard algorithm lost the board at around 3m. At this point the 2.5cm squares on the board were only around 3 pixels in height and so little more in terms of performance could be expected from the algorithm. The colour detection operated up to 10 m at which point the marker was so small on the camera that it was removed as noise during opening. All three algorithms only operated over a relatively short distance in comparison to how far the kart might be from the marker. This would mean that the actual marker to be tracked would need to be significantly larger then those used in testing and/or a higher resolution camera would have to be used.

TABLE I
DISTANCE BEFORE DETECTION OF MARKER IS LOST (ALL VALUES IN METERS).

Algorithm	Run 1	Run 2	Run 3	Average
Colour	10	11.5	9	10.2
SURF	1.0	0.7	1.1	0.9
Chessboard	3.5	3.0	3.2	3.2

E. Robustness to background noise

The ability of the algorithm to operate on a variety of backgrounds was tested. How much the colour detection was effected was closely coupled to how well it performed in varying light. If the range of hues and saturations was set to be very close to the markers colour then almost no issues with background objects occurred. An example of this can be seen in Figure 1 where the blue ball is detected while sitting next to a blue can and a blue towel with neither of these being detected. These close tolerances meant that lighting had to be kept almost constant for the marker to be detected. If these ranges were relaxed to work in more varying light then any similarly coloured part of the background would also be detected. The SURF algorithm also had issues with the background. When the image was being tracked a small amount of the points (usually less than 10%) would be attributed to points in the background. These incorrect points did not affect the algorithm however as the RANSAC algorithm detected them for the outliers that they were. The main issues came when the image was far away, partly obscured or absent from the scene. In these situations a large number of points would be detected at locations in the background giving false positives for locations of the marker. These locations were almost always easy to detect as they usually claimed that the marker was hundreds of meters away however they still required filtering and subtracted from the robustness of SURF. The chessboard was almost completely unaffected by the background. This was because while on occasion a false corner point was detected as it did not match the structure of the rest of the points it was discarded without problem. This lead to no false positives as for this to occur 42 false points would have had to of been detected laid out in a 6 by 7 grid.

F. Accuracy of distance measurement

The accuracy of the distances given was tested by placing the marker at 3 different points and recording the error of the system. These measurements are shown in Table II. The SURF detection performed poorly in this test having an average error of 18%. It also failed to register on the 2 meters test performing easily the worst of the three. Chessboard detection was almost as accurate as the tape measure being used to compare the distances giving a maximum error of only 1.5%. Colour detection performed far better than expected given its

TABLE II
ACCURACY OF DISTANCE MEASUREMENTS

Algorithm	At 0.5m		At 1m		At 2m		Avg
	Value	Error	Value	Error	Value	Error	
Colour	0.57	14.0%	1.03	3.0%	2.07	3.5%	6.8%
SURF	0.61	22.0%	1.14	14.0%	-	-	18.0%
Chessboard	0.495	1.0%	0.99	1.0%	1.97	1.5%	1.2%

crude distance calculation method having an error of only 6.8%. Something that must be taken into account however is that the colour detection is in reality less accurate then this test indicated. This is because the amount of the marker detected depends on the lighting conditions and the distance was calculated using size of the marker. This meant turning on a light could change the distance the process believed the marker was at by a large amount (20% in one rough test).

G. Testing conclusions

From these tests it was concluded that chessboard detection was the only algorithm that was appropriate for use on the kart. This was because SURF was too sensitive to motion blur and had too short a range for the application. While colour detection out performed chessboard detection in several areas its sensitivity to light changes meant that it would never be robust enough to be useful.

VI. PATH FOLLOWING

Once the kart had located a marker a method for following the marker had to be implemented. The simplest solution of driving towards the marker had many drawbacks. An example would be that if the kart was following the marker and the marker moved forwards, then after a few meters turned to the right. The following kart would cut the corner and potentially smash straight into an obstacle. A second problem would be that if the marker left the karts field of vision the kart would be forced to immediately come to a stop as it no longer had a location to drive towards.

To solve these problems an algorithm was developed that would attempt to follow the path the kart had taken. This algorithm utilized the marker position given by the computer vision and combined it with information from the karts speed sensor and the position of the steering wheel. The program used the karts sensors and the time at which each reading was taken to calculate an estimate for its absolute position and orientation. Once the location of the kart had been found the relative position and orientation of the chessboard marker if present was also recorded. After this had been done the absolute position of the marker was calculated. The current and all previous positions were stored in an array.

Armed with these absolute position estimates the core of the path following algorithm was very simple and executed as follows.

1) The kart would select the first point on the path
2) The kart would set the steering to point at the point and drive towards it
3) When the kart came within a minimum distance of the line (set to the turning radius of the kart) or the point became too "old" the kart would deem itself to have reached the point
4) The kart would then repeat the process for the next point

By setting the criteria for meeting the point to the turning radius of the kart this meant that the kart would always be able to turn to be within this distance of the next point regardless of where it lay. The kart skipping a point if it became older then a set value (in testing set to 10 seconds) meant that if the kart began to lag behind the object it was tracking it would start cutting corners in order to catch up to the target. This meant that if the target was slowly moving through tight turns the system would work on matching its path and when the target was moving quickly the system would priorities keeping up with the target. The time limit also helped to remove oscillations that could be set up when the kart was following a line.

The method used to estimate the absolute position was prone to compounding errors. This problem was slightly mitigated in that error could be present in the position of the kart as long as the same error was present in the recorded positions of the marker. This meant that the only error that would matter would be that introduced in the time between the marker being recorded at a point and the kart reaching that point. This meant when the kart was closely following a marker the time over which error could compound would be limited to a few seconds. In most environments the kart might navigate an error of the order of a meter could be acceptable. This meant that the position estimation could be very crude and still produce reasonable tracking.

VII. PERFORMANCE OF PATH FOLLOWING

Due to delays in the project converting the go-kart to a point where it could be interfaced with the laptop the algorithm was never tested on the kart. To allow for some basic testing a simple GUI output was instead added to the program. This mapped the absolute location of the kart and the chessboard as well as the path they had taken. It also highlighted the point which the kart was currently driving towards. Fake input from the steering and speed sensors were created giving the kart a constant speed and a maximum turn rate of 30 degrees per second. This system is shown in Figure 3. This interface allowed the system to be debugged and tested though was fairly limited in comparison to the actual go-kart. Its largest problem was that as the system believed that it was moving forward and turning even though it was in fact stationary because of this it saw the marker it was tracking in front of it as moving to maintain the same relative position. This meant that testing the system's ability to follow patterns laid out by the marker was seriously hindered.

VIII. FUTURE WORK

Time constraints on this project meant that not all of the desired functionality was explored and implemented. One area that needed further work was what the path following

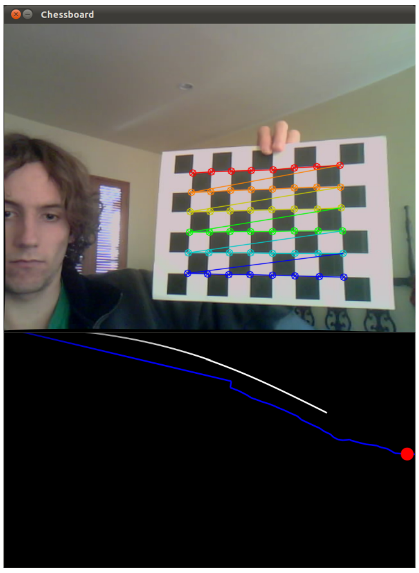

Fig. 3. Path finding GUI using chessboard detection. The path of the kart is shown in white, blue for the chessboard and the red dot is the current point the kart is driving towards

algorithm did if it had lost sight of the marker and had not relocated it by the time it came to the end of its line of remembered points. This situation could occur if the vehicle it was following was performing a close turn around an obstacle or if it went down into a depression and out again. A possible solution to this would have been to fit a spline to the points the system had been following and extrapolate out from this the most likely coarse the leader with the marker had taken.

A second possibility that was not followed was the idea of using multiple methods of detection to find the marker. Chessboard detection worked well on its own but its performance suffered during motion blur, had some issues at large distances and had a slow output rate when it lost the target marker. Colour detection had none of these same problems but suffered from far more crude distance measurements and the need to be recalibrated for different light conditions. The strengths of the two systems could have been combined by using the chessboard system to constantly recalibrate the colour detection system using a brightly coloured chessboard instead of a black and white one to act as the marker for both systems. This would mean that the colour detection would have no longer had the issues with lighting and distance location as every time the chessboard was detected it would be used to reset the colour being tracked and to adjust the distance readings so the two systems matched.

IX. Conclusion

Three computer vision solutions were looked at for tracking a marker for use in allowing a kart to follow a lead vehicle. Colour detection was found to be light weight, robust to movement and reasonably accurate in constant lighting (error of less than 7%). It was however plagued by a large sensitivity to lighting changes that made it unusable as a robust marker location system. The SURF algorithm ran slowly and its inability to handle distances over 1m and even small amounts of motion blur made it inappropriate for the application of being mounted to a moving vehicle. Chessboard detection was the only system that was robust enough to detect a marker and give its position reliability. It was slightly hindered by it inability to deal with large amounts of motion blur and large distances. The output of these systems were combined with sensors measuring the speed and wheel angle of the kart to produce a system that followed the path the marker had taken rather than just driving at the marker giving an accurate following method that could in the tests conducted effectively follow a marker placed on a leading vehicle. This following method was never tested on the kart however due to the karts drive by wire systems not being finished before the end of the project.

References

[1] R. A. Lewis and A. R. Johnston, *A Scanning Laser Rangefinder for a Robotic Vehicle*, 1977, vol. 2, pp. 762–768. [Online]. Available: http://adsabs.harvard.edu/abs/1977STIN...7722467L

[2] T. Leyvand, "Kinect identity: Technology and experience," *Computer*, vol. 44, no. 4, pp. 94–96, 2011. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5742015

[3] J. Hightower, R. Want, and G. Borriello, "Spoton: An indoor 3d location sensing technology based on rf signal strength," *Communications*, vol. 2000-02-02, no. 2000-02-02, pp. 1–16, 2000. [Online]. Available: http://seattle.intel-research.net/people/hightower/pubs/hightower2000indoor/hightower2000indoor.pdf

[4] S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the hsv color space for image retrieval," in *in International Conference on Image Processing (ICIP). 2002: p. 589-592. VIIth Digital Image Computing: Techniques and Applications, Sun C., Talbot H., Ourselin*, 2002, pp. 589–592.

[5] D. Prochzka and T. Koubek, "Augmented reality implementation methods in mainstream applications," *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis*, vol. LIX, no. 4, pp. 257–266, 2011.

[6] A. De La Escalera and J. M. Armingol, "Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration," *Sensors (Peterboroug*, vol. 10, no. 3, pp. 2027–2044, 2010. [Online]. Available: http://www.mdpi.com/1424-8220/10/3/2027/

[7] H. Bay, T. Tuytelaars, L. Van Gool, and L. Van Gool, "Surf: Speeded up robust features," *Computer VisionECCV 2006*, vol. 3951, no. 3, p. 404417, 2006. [Online]. Available: http://eprints.pascal-network.org/archive/00002183/

[8] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," *Proceedings International Conference on Image Processing*, vol. 1, no. 2002, pp. I–900–I–903, 2002. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1038171

[9] R. Green, "Cosc428 lectures," registration lecture, University of Canterbury.

[10] K. G. Derpanis, "Overview of the ransac algorithm," *Image Rochester NY*, vol. 4, pp. 2–3, 2010. [Online]. Available: http://www.cs.yorku.ca/~kosta/CompVis_Notes/ransac.pdf

[11] G. Bradski, "The opencv library," *Dr Dobbs Journal of Software Tools*, vol. 25, no. 11, pp. 120–126, 2000. [Online]. Available: http://opencv.willowgarage.com

[12] V. Douskos, I. Kalisperakis, and G. Karras, "Automatic calibration of digital cameras using planar chess-board patterns," *Camera*, pp. 132–140, 2006. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.78.4664&rep=rep1&type=pdf

[13] H. Schaub and C. E. Smith, *Color snakes for dynamic lighting conditions on mobile manipulation platforms*, 2003, vol. 2, pp. 1272–1277. [Online]. Available: http://ieeexplore.ieee.org/ielx5/8832/27959/01248820.pdf?tp=&arnumber=1248820&isnumber=27959