Defect analysis of grit-blasted or spray-painted surface using Vision Sensing Techniques

G. Sen Gupta[*, +], Tin Aung Win[^], Chris Messom[#], Serge Demidenko[*], Subhas Mukhopadhyay[*]
[*]IIS&T, Massey University, Palmerston North, New Zealand
Email: {g.sengupta, s.demidenko, s.c.mukhopadhyay@massey.ac.nz}
[+]School of Electrical and Electronic Engineering, Singapore Polytechnic, Singapore
[^]Singapore Maritime Academy, Singapore Poly, Singapore, Email: win@sp.edu.sg
[#]II&MS, Massey University, Albany, New Zealand, Email: c.h.messom@masey.ac.nz

Abstract

The paper discusses a vision-based framework for the analysis of defects on a grit-blasted or spray-painted metal surface. The system employs commodity, off-the-shelf hardware and tailor-made software, with user-friendly interface, making it cost effective yet accurate. A picture of the surface (1mx1m) is captured by a colour CCD camera and processed by a Frame Grabber card at a resolution of 320x240 pixels. Presented in the paper is a blob identification technique, using a two-pass sequential algorithm, which is used to locate the defects. Closely spaced defective blobs are merged and area-thresholding is done to eliminate noise. To desensitize the system to variations of light intensity, the YUV colour space model is used. The system reports the 'outer extent', coordinates of the centre and the area of up to 5 defective spots. The processing is fast and is done in real time. The system has been tested indoors under varying light intensity. Successful field trials have been conducted at the Jurong Shipyard, Singapore.

Keywords: Grit-blasting robotic-arm, blob identification, area-thresholding, defect analysis, LUT

1 Introduction

When ships come to the dry dock yards for repair, one of the major work done on its outer surface is high pressure blasting with grit to remove rust, paint and sludge. The surface is then spray coated with fresh paint. The entire process is manual, expensive and environmentally hazardous. The ship repair industry requires an *inexpensive* system to automate this entire process to increase worker safety, reduce cost and minimise the wastage of grit and anti-fouling paint, thereby increasing environmental compliance. Additionally, the system must be user friendly and easy to operate requiring minimal effort to train the software to recognise colours.

The grit-blasting equipment developed by our team is shown in Figure 1. In the blasting process, there are often pockets where the cleaning is below standard as shown in Figure 2. In the manual process, the operator directs the hose to the defective area and re-blasts it. The same is true for spray painting. However, in an automated process, we need a visual inspection system to locate the defective areas so that the blasting hose (or the spray painting gun) can be automatically repositioned to re-blast (or repaint). Self Organising Map (SOM) algorithms are currently employed for surface inspection [1], which are slow as compared to the technique proposed here.

Since the blasting or painting is done using a mechanised X-Y positioning robotic arm, the visual inspection system must calculate the 'outer extent' of

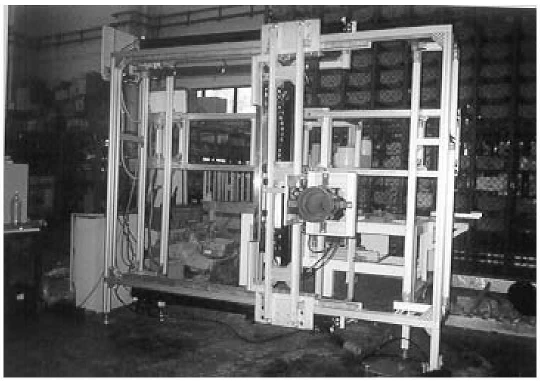


Figure 1: Grit-blasting robotic-arm

the defective blob so that the defective spots can be redone. The centre of a blob and its area is calculated for statistical data and process analysis. Image segmentation and colour thresholding is done to find the size, position and extent of the defective blobs. Colour training of the software is done in the YUV colour space as it gives the advantage of segregating the luminance band from the chrominance band. This makes the vision inspection system more robust in the face of varying light intensity. A fast YUV colour look-up table (LUT) mechanism has been implemented to greatly improve the processing speed.

The system has wide ranging applications not only in the ship repair industry, but also in the general sheet metal fabrication industry where automated vision inspection is required. Since the vision system is based on commodity hardware, it is cheap, yet effective.

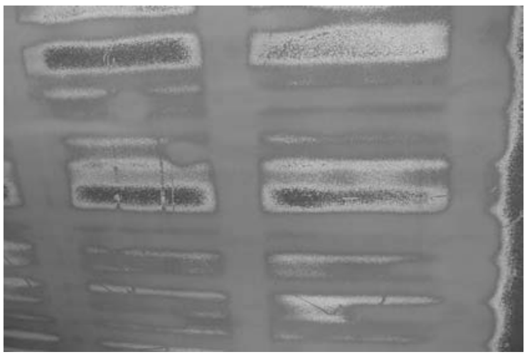


Figure 2: Below-standard blasted surface

2 System Hardware Description

The essential system components are the manipulator, the grit-blasting nozzle with suction cone, spray gun for painting and the vision inspection hardware. These are described in the following sub-sections.

2.1 The manipulator

The robotic arm (manipulator) is essentially an X-Y-Z positioning system carrying one grit-blasting nozzle and one spray-painting nozzle. It is mounted on a cherry picker as shown in Figure 3. The motion is controlled by pneumatic cylinders to carry out all X-Y-Z movements. Pneumatic breaking is used for precise point positioning. The stroke distances for X, Y and Z axis are 150cm, 150cm and 30cm respectively with positional accuracy of 5mm and variable speed of 0.1 to 0.3m/s in any axis. The operator can manually control the position and movement of the nozzles or program it to follow a pre-defined trajectory.

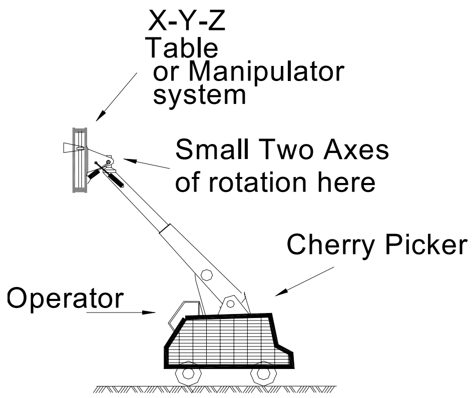

X-Y-Z Table or Manipulator system
Small Two Axes of rotation here
Cherry Picker
Operator

Figure 3: Manipulator on a cherry picker

2.2 Grit Blasting Hardware

The grit-blasting nozzle spouts grit, mixed with air, at high pressure to strike the surface to be cleaned. The nozzle is enclosed within a vacuum cone which is used to suck back the grit, which would otherwise fall to ground, and dust. This pollutes the atmosphere less and the grit which is collected can be reused.

2.3 Spray-Painting Hardware

It consists of a spray gun which mixes paint with pressurised air before discharging through a nozzle. The nozzle comes in different bore size and can be changed for different paint viscosities. The air pressure can also be altered.

2.4 Vision Inspection Hardware

The vision inspection system consists of a Pulnix 7EX NTSC colour CCD camera and a FlashBus MV Pro frame grabber card which is plugged into a PC. The camera is mounted by the side of the X-Y-Z positioning arm at a fixed distance of 1m from the inspection surface. The field of view of the camera is an area of 1m x 0.75m. To protect the camera from dust and paint, it is housed in an air-tight chamber with clear glass in front.

The image is captured at a resolution of 320x480 pixels at a sampling rate of 30Hz. The odd and even fields of the interlaced image are processed separately. Hence the effective image resolution is 320 x 240 delivered at a sampling rate of 60Hz. Each pixel covers an area of 3.125mm x 3.125mm. The captured image is processed using a Pentium II PC (450MHz, 128MB RAM). The frame grabber card was configured for off-screen image capture, as shown in Figure 4.

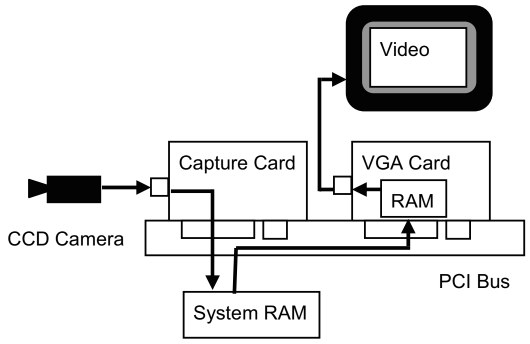


Figure 4: Image capture in off-screen capture mode

3 Operation Sequence

After a surface section has been blasted (or painted), the manipulator moves to the next section such that the camera now looks at the section that has just been blasted (or painted). Once the surface has been visually inspected and no defects are detected, the blasting (or painting) starts for the new section. However, if defects are detected, the manipulator is positioned to re-blast (or re-paint) the defective areas of the previous section. While the image is captured and analysed, the blasting (or painting) is put on hold so as not to affect the image quality due to vibrations in the manipulator structure.

4 Blob Detection using Color thresholding

Several techniques for colour component identification have been presented in [2]. A blob detection technique has been successfully employed to track moving objects [3] with colour patterns. In the proposed method for visual inspection, the software is trained to recognise the background colour – the paint colour in the case of spray painting - and the YUV thresholds are defined. In the calibration phase, the lower and upper limits of Y, U and V ranges may be manually tweaked for best results under varying light intensities. On initiation of inspection, it searches through the image, testing if a pixel *does not* belong to the calibrated background colour. The 'non-member' pixels are then grouped together to create the 'defective' colour patches. Component labelling uses the sequential algorithm, which is a two-pass labelling technique. The labels are identifiers that are incremented starting from the value of 1. Ideally the number of labels used is equal to the number of 'defective spots' in the image.

The procedure of testing the membership and grouping the pixels is made of 5 steps combined into two passes.

A. FIRST PASS (Steps 1 to 4)

1. Scan the image in the incremental window from left to right, top to bottom
2. If the pixel in the image is within the YUV threshold values of the colour of interest, then
   (a) If only one of its upper and left neighbours has a label, then copy the label.
   (b) If both upper and left neighbours have the same label, then copy that label.
   (c) If both upper and left neighbours have different labels, then copy the upper pixel's label and enter the labels in an equivalence table as equivalent labels.
   (d) Otherwise assign a new label to this pixel and enter this label in the equivalence table.
3. Check if there are more pixels to consider then go to step 2, otherwise proceed to step 4.
4. Find the lowest label for each equivalent set in the equivalence table – prepare the equivalence table

B. SECOND PASS (Step 5)

5. Scan the picture. Replace each label by the lowest label in its equivalent set.

Figure 5 shows a representation of the binary image where the 0's represents the background of the image and the 1's represent the objects of interest, the defective spots. It can be seen that there are two objects of interest in this image, one in the left part and the other in the right.

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	0	1	0
0	1	1	1	1	0	0	0	0	1	1	0	1	0
0	1	1	1	1	1	1	0	0	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5: The binary Image

Figure 6 shows the image after the first pass of the algorithm. Different objects have different labels. However due to a weakness in the first pass of the algorithm some of the objects have multiple labels associated with them, for example the object on the left with labels 3 and 1 and the object on the right with the labels 2 and 4.

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	0	2	0
0	**3**	1	1	1	0	0	0	0	**4**	**4**	0	2	0
0	**3**	1	1	1	1	1	0	0	**4**	**4**	**4**	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6: The image after the first pass

Figure 7 shows the image after the second pass of the algorithm, which resolves the problem of multiple labels for single objects. Having identified the separate objects, the centre of an object is calculated using a centre of gravity calculation.

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	0	0	2	0
0	1	1	1	1	0	0	0	0	2	2	0	2	0
0	1	1	1	1	1	1	0	0	2	2	2	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7: Image after second pass

5 Computing Area, Centre and extent of defective spots

5.1 Area thresholding and spot merging

After all the 'defective spots' have been identified and labelled and before their location is computed, area thresholding is done for two reasons – very tiny spots, below a certain threshold, are discarded as noise and spots which are very close to each other are merged. When closely spaced defective spots are merged, the re-painting (or re-blasting) process becomes more efficient as the spray head has to travel to the 'merged defective spot' only once. The noise limiting threshold and the minimum separation distance for defective spots to be merged can be altered from the applications GUI.

5.2 Locating the defective spots

For a binary image of m x n pixels, the area A and position $(\bar{x}, \bar{y})$ is calculated using equation (1) and equation (2) respectively (zero-order moment and centre of gravity).

$$A = \sum_{i=1}^{n} \sum_{j=1}^{m} B[i,j] \qquad (1)$$

$$\bar{x} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} jB[i,j]}{A} \text{ and } \bar{y} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} iB[i,j]}{A} \qquad (2)$$

where B[i,j] represents the binary image.

In the second pass, when each label is replaced by its lowest equivalent, the *extent* of the component (*MinXco, MaxXco, MinYco* and *MaxYco*) is also calculated.

6 A fast LUT access

6.1 Dealing with RGB colour space

The blob detection algorithm can be implemented on commodity hardware for visual inspection of surfaces. The image digitization is done using the FlashBus MV Pro frame grabber card which provides pixel colour information in RGB. A convex partition of the RGB color space created for each colour identifier, as shown in Figure 8, was explored [4]. This convex partition is specified by a range of RGB values namely, R_{min} - R_{max}, G_{min} - G_{max}, B_{min} - B_{max}.

Blob identification based on RGB colour space was not found to be useful and reliable in the face of varying light intensities as the luminance could not be separated from chrominance. Instead a convex colour subspace, defined in the YUV colour space was implemented with greatly enhanced robustness, as Y corresponds to light intensity component of the colour.

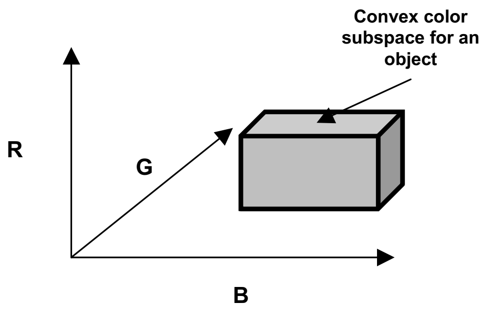


Figure 8: Convex color subspace

6.2 Defining YUV thresholds

To define the YUV colour subspace, a sample of the image is captured and the colour of interest is zoomed in. In the zoomed image a rectangular region is defined, within which, each pixel is processed to calculate its YUV value using the colour space transformation matrix (3).

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 02.99 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{aligned} U &= 0.565(B - Y) \\ V &= 0.713(R - Y) \end{aligned} \qquad (3)$$

From the computed YUV values, the *MinY, MaxY, MinU, MaxU, MinV* and *MaxV* are set. A user may manually fine-tune these thresholds using the application's GUI. Usually a wider range of Y values are desirable giving it more bandwidth to account for varying light intensity.

6.3 Membership testing

For the two-pass sequential algorithm for blob-detection, each RGB colour pixel of an image needs to be tested to determine its colour sub-space membership. Hence the mechanism used for thresholding warrants close scrutiny and requires careful efficiency consideration. Since the colour boundaries are defined in YUV colour space, each pixel would have to be converted from RGB to YUV, using equation (3), before testing membership using equation (4). This is computationally very intensive and the processing will be very slow.

```
IF ( (Y >= MinY) AND (Y <= MaxY)
     (U >= MinU) AND (U <= MaxU)
     (V >= MinV) AND (V <= MaxV) )
THEN pixel_of_interest = TRUE                    (4)
```

Equation (3) requires 5 multiplications and in equation (4), it may require up to 6 conditional branches to determine a pixel of interest. To improve efficiency, implementations using Boolean valued decomposition of the multidimensional threshold have been tested [5]. This, however, still requires the colour space to be transformed from RGB to YUV.

6.4 Colour Look-Up-Table (LUT)

Our implementation uses a large one dimensional colour look-up-table (LUT) and an indexing technique based on the RGB value of the pixel. The index is created, using equation (5), which is used to access the LUT.

$$\text{index} = R*65536 + G*256 + B \qquad (5)$$

For a 24-bit RGB colour output from the frame grabber card, the maximum value of R, G or B is 255. Thus the size of the LUT is 256x256x256 bytes (16MB).

6.4.1 Posting the LUT

Once the YUV thresholds have been defined for each colour, the LUT is posted with colour identities (IDs) for the entire RGB colour space as shown in Figure 9.

```
for (r=0; r<256; r++)
 for (g=0; g<256; g++)
  for (b=0; b<256; b++)
   {
    y=(299*r+587*g+114*b+500)/1000;
    u=(565*(b-y) + 128000)/1000;
    v=(713*(r-y) + 128000)/1000;
    index = r*65536 + g*256 + b;

    //-- initialise on update --
    LUT[index] = NoCOL;

    //-- Reference Colour range --
    if ( (MinY<=y && y<=MaxY) &&
         (MinU<=u && u<=MaxU) &&
         (MinV<=v && v<=MaxV))
    {
     LUT[index] = RefCOL;
    }
  }
```

Figure 9: Posting the LUT with colour ID

The time it takes to update the LUT is not of any consequence as it is done during the colour tuning phase. It is important that during the inspection time, the processing should not take unduly long and hence repeated multiplications and logical ANDing must be avoided.

6.4.2 Inspecting the LUT

To test whether a pixel is in the YUV sub-space, given its RGB value, the index is calculated using equation (5) and the LUT content at that indexed location is tested as shown in Figure 10.

To further improve the processing speed, the multiplications in equation (5) were replaced by shift-left operations as in equation (6).

$$\text{index} = R*<<16 + G*<<8 + B \qquad (6)$$

```
index = r*<<16 + g*<<8 + b;
if ( LUT[index] == RefCol )
//-- it is a desired pixel
{
  //-- process the pixel
}
```

Figure 10: Inspecting the LUT

7 Experimental results

The defective colour spots on a given background were simulated in the lab environment and the system thoroughly tested before the field trials.

The vision system and algorithms were tested in two phases. In the first phase, a piece of actual grit-blasted metal plate was inspected in controlled light conditions in the lab. The second phase of tests was carried out in Jurong Shipyard, Singapore under actual operating conditions.

The outer boundaries of the defective spots, are calculated with an accuracy of ± 3mm. Light intensity variation from 600 to 1100 lux have been tested with no deterioration in the accuracy and repeatability of measurements. Defects, as small as 5mm x 5mm, were detected.

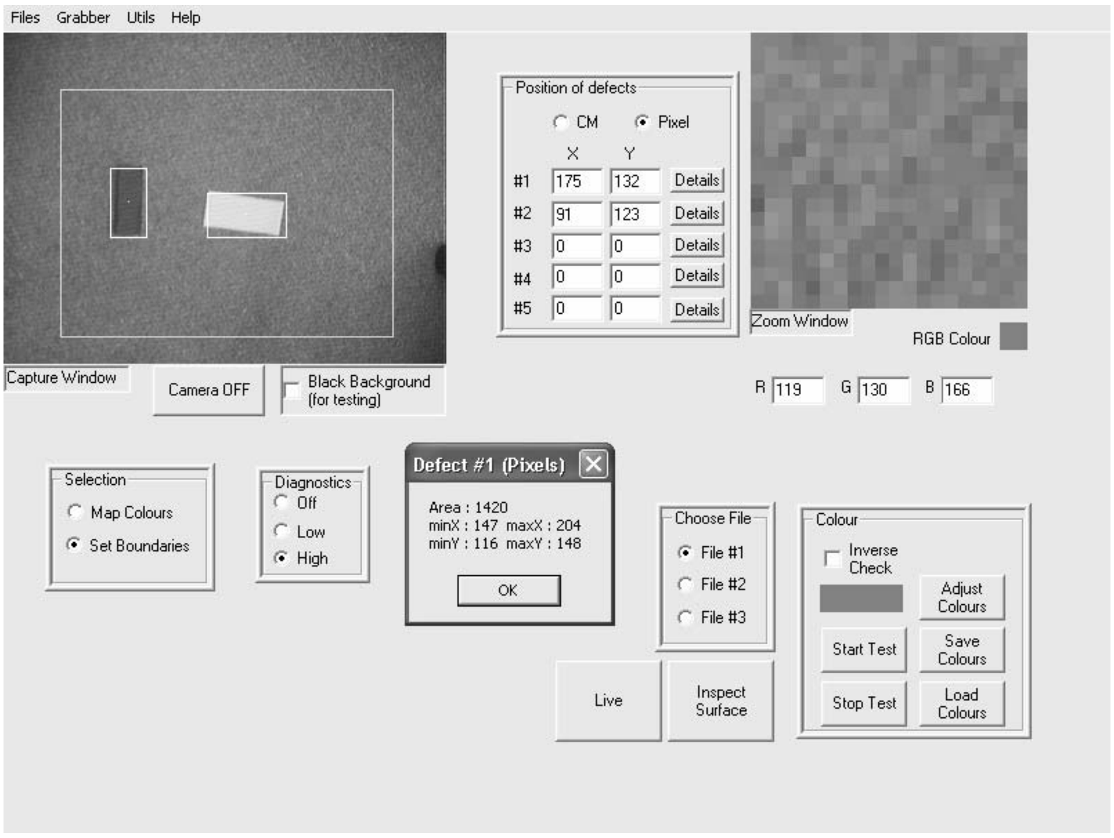
The entire picture is analysed and data calculated in 13ms which is well within the frame refresh rate of 60 Hz. The technique presented here compares favourably to the color threshold based approaches discussed in [6]. The drawback of the proposed LUT is, its size being huge, it occupies a substantial chunk of system RAM. This tends to slow down processing on processors which do not have large L2 cache like Pentium III PCs as it results in frequent disk file swapping.

The graphical user interface is shown in Figure 11. The system can be trained to recognise 3 different background colours.

Future work involves processing the image at a resolution of 640 x 480 to enhance the accuracy with which the outer boundaries are calculated and to detect even smaller defects. The vision inspection system has to be integrated with the grit-blasting (and spray-painting) control mechanism so that after the location of the defects are known, the detective areas can be automatically re-blasted (or repainted).

8 Conclusions

The paper presented efficient algorithms for blob identification using fast access to colour look-up-table. The system is inexpensive as it is based on commodity hardware. The accuracy of detection is



Position of defects: ( ) CM [✓ Pixel]

	X	Y	
#1	175	132	Details
#2	91	123	Details
#3	0	0	Details
#4	0	0	Details
#5	0	0	Details

R 119 G 130 B 166

Black Background (for testing): ☐

Selection: ( ) Map Colours [✓ Set Boundaries]

Diagnostics: ( ) Off ( ) Low [✓ High]

Defect #1 (Pixels)

```
Area : 1420
minX : 147  maxX : 204
minY : 116  maxY : 148
```

Choose File: [✓ File #1] ( ) File #2 ( ) File #3

Colour: Inverse Check ☐

Figure 11: User Interface depicting two defects on simulated defective surface

very high and works under varying light intensities. The system finds application not only in ship repair industry but, in general, in any sheet metal fabrication process where automated inspection of surface is required.

9 Acknowledgements

The project was entirely funded by Tote Board, Singapore. The development and research was done in Singapore Polytechnic in the laboratory facilities provided by Singapore Maritime Academy (SMA) and School of Electrical and Electronic Engineering (SEEE). Site testing facilities were provided by Jurong Shipyard, Singapore

10 References

[1] M..Niskanen, O. Silven and H.Kauppinen, “Experiments with SOM Based Inspection of Wood”, *International Conference on Quality Control by Artificial Vision (QACV 2001), Le Creusot*, pp 311-316 (2001).

[2] M. Ramesh Jain, R. Kasturi, and B.G. Schunck, *Machine Vision*, McGraw-Hill International Editions, Computer Science Series, International Edition (1995).

[3] G. Sen Gupta, C.H.Messom, S.Demidenko and Lim Yuen Siong, “Identification and Prediction of a Moving Object Using Real-Time Global Vision Sensing”, *IMTC2003, Vail, Colorado*, pp 1402-1406 (2003).

[4] C.H.Messom, S. Demidenko, K. Subramaniam and G. Sen Gupta, “Size/Position Identification in Real-Time Image Processing using Run Length Encoding”, *IMTC2002, Anchorage, USA*, pp 1055-1060 (2002)

[5] J. Bruce, T. Balch and M Veloso, “Fast and Inexpensive Color image segmentation for Interactive Robots”, *IROS 2000, San Francisco*. pp 2061-2066 (2000)

[6] J.Baltes. "Practical camera and colour calibration for large rooms", *RoboCup-99: Robot Soccer World Cup III*, New York, pp 148-161 (2000)