Movement and Color Detection of a Dynamic Object An Application to a Mobile Robot

Roman Osorio[1], Sinuhé García[1], Mario Peña[1], Ismael Lopez-Juarez [2], Gastón Lefranc[3]

[1] Departamento de Ingeniería de Sistemas Computacionales y Automatización,
Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas,
Universidad Nacional Autónoma de México,
roman@servidor.unam.mx

[2] CINVESTAV, Grupo de Robótica y Manufactura Avanzada, México.
ismael.lopez@cinvestav.edu.mx

[3] Pontificia Universidad Católica de Valparaíso,
Avda. Brasil 4950, Valparaíso Chile,
glefranc@ucv.cl

Abstract: The paper describes the integration of several image processing algorithms necessary to recognize a particular color and the movement of an object. The main objective is to detect the object by its color and track it by a mobile robot. Mean filter is applied to soften and sharpen the input image. Then, RGB filter is applied to calculate the center of mass and area of the object and to locate its position in a real environment to develop the robot motion. These algorithms are applied to a mobile robot, in a tested scenario, tracking an object.

Keywords: Image processing, color recognition, mobile robot.

1. Introduction

Imaging methodology for movement detection of dynamic objects in mobile robot applications is presented using an optical color camera and computer vision algorithms by way of integrating all elements of hardware and software as a one component.

Main objective of the developed methodology for finding dynamic objects is to integrate different computer imaging vision algorithms implementing filtering, features and center of mass extraction techniques in real time for finding and tracking moving objects.

One of the utilities of images processing is currently the implementation of vision systems in mobile robots which by natural effect include one or two cameras. If the system is stereo vision, cameras can obtain images of their environment and process them [10]. It is possible to obtain important information in the image processing; the robot actions are related according to these data. One of the applications of the image processing implementation is the follow-up of a colored object via a robot [1], [2]. There exist different methods to detect movement of a dynamic object [9], [15] - [19]. Video-based moving objects detection approach is a research area related to image processing, pattern recognition and artificial intelligence. Real-time moving object detection is the premise of the video tracking and analysis, which has great theoretical value and practical value. One of these values is to obtain a sequence of video and then motion detection and motion segmentation. It can be made by background modeling and subtraction and then processing the images to have a mask of the object to follow [9]. The shape-based approach is often insufficient especially in case of large data sets [12]. Another object recognition approach is to use color (reflectance) information. It is well known that color provides powerful information for object recognition, even in the total absence of shape information. A common recognition scheme is to represent and match images on the basis of color invariant histograms [7], [13], [14]. A work that proposes a tracking system for moving objects with specified color and motion information uses color transformation and AWUPC computation is given by [16]. Other work utilizes a framework for carrying object detection using kinematics manifold embedding and decomposable generative models by kernel map and multilinear analysis [17]. The color-based matching approach is widely in use in various areas such as object recognition, content-based image retrieval and video analysis. When the applications is that a robot follow-up of a colored object, the robot can perform several moves: forward, backward, right/left turn. Robot SRV1-blackfin camera used in this research takes images of frames resolution of 160x120, 320x240, and 640x480. By using an image filter it is possible to

interpret the captured images, and to obtain relevant information for mobile robot [3]-[6]. One of the most successful algorithms is based on the Scale Invariant Feature Transform (SIFT) [21]. SIFT has been integrated into a number of commercial products, including Sony's Aibo, Bandai's NetTansor robots, and the visual Simultaneous Localization and Mapping (vSLAM) system [20] by Evolution Robotics. An unsupervised algorithm to learn object color and locality cues from the sparse motion information is published. First detects key frames with reliable motion cues and then estimates moving sub-objects based on these motion cues using a Markov Random Field framework. From these sub-objects, it learns an appearance model as a color Gaussian Mixture Model [19].

SIFT is a method that recognizes multiple objects and query images based on minimal training input. It's simple to use: no specialized expertise, dataset, nor equipment is required to train new models. Discrimination between multiple learned objects is handled efficiently. It's completely robust to changes in scale and to in-plane rotations, and it accommodates mild perspective distortions that arise from out of-plane rotations [21]. In the acquired image there are many defects and noise, but not all are common and some people have develop a real time systems for inspection, detection and tracking moving objects in production systems such as potato inspection where the potatoes are inspected (size and color) on the fly while passing on a belt conveyor [25]. A machine vision system trained to distinguish between different objects of the same class but with different characteristics uses threshold techniques for image segmentation [26]. A Neural Network and a Vector Description methodology to recognize and calculate POSE of manufacturing objects uses a Color classification method using image processing techniques [27].

A new algorithm of moving object detection is proposed. The moving object detection and orientation uses a pixel and its neighbors as an image vector to represent that pixel modeled different chrominance component pixel as a mixture of Gaussians. In order to make a full use of the spatial information, color segmentation and background model were combined. Simulation results show that the algorithm can detect intact moving objects even when the foreground has low contrast with background [18].

In this paper, the detection of moving object is made by using its color. The object is tracked utilizing a mobile robot. The Mean filter is applied to soften and sharpen the input image and the RGB filter for color detection is applied. Later on, the center of mass and area of the object is calculated. These algorithms are applied to a mobile robot, in a tested scenario, tracking an object.

2. Mean Filter

Because there may be noise in the image, it is necessary to use a filter, such as the Gaussian smoothing filter, The Median filter, the Mean Filter, etc.[7], [11]. The Median Filter computes the median of the pixel´s surrounding pixel´s values. This filter has the disadvantage of being slower, requiring more processing. The Mean Filter replaces each pixel value in an image with the mean ('average') value of its neighbors, including itself (the analyzed pixel). This has the effect of eliminating pixel values which are no representative of their surroundings. Mean filter is considered as a low pass filter according to the definition only allowing the entry of low frequencies and attenuating the higher ones, reducing the spatial intensity derivatives present in the image. This effect operates as 'softening' the image. The Mean filter is also considered as a convolution filter, based on a kernel, which represents the shape and size of the sampling area to calculate the arithmetic mean or average. Usually a 3 x 3 matrix is used for this kernel. The convolution is a simple mathematical operation essential for many common image processing operators and provides a way to multiply two arrays of numbers of different size but of the same dimension. The kernel size depends on the intensity of the desired smoothing and sharpness. With a small kernel size, smoothness will be low and sharpness high; on the other hand, if the size of the nucleus is large, smoothness will be high and sharpness low, so images are blurred.

$$K = \frac{1}{k}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} \quad (1)$$

k is the number of elements in the matrix and *K* is the matrix that represents the core to be convoluted.

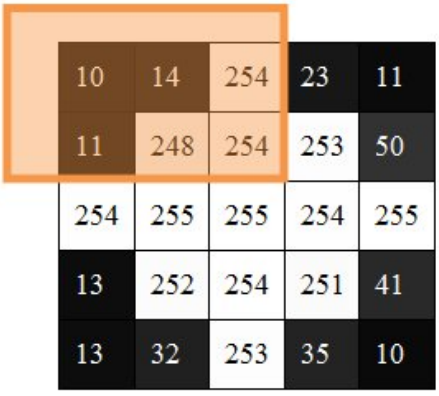
(-1,-1)	(-1,0)	(-1,1)
(0,-1)	(0,0)	(0,1)
(1,-1)	(1,0)	(-1,1)

Using an array I which represents the image and a matrix *K* which will make the convolution, an array O is obtained. The following matrix I represents a gray scale image.

I =

10	14	254	23	11
11	248	254	253	50
254	255	255	254	255
13	252	254	251	41
13	32	253	35	10

In the case of the edges, not available indexes are considered as 0 values. For example to calculate O_{00}, the values are shown in (2).



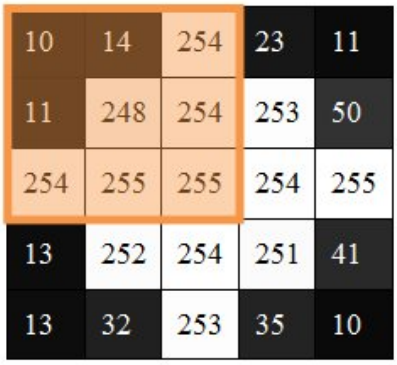
$$O_{00} = (0 + 0 + 0 + 0 + 10 + 14 + 0 + 11 + + 248)/9 = 31 \quad (2)$$

To calculate the value of O_{11}:

$$O_{1,1} = (I_{0,0} * K_{0,0} + I_{0,1}*K_{0,1} + I_{0,2}*K_{0,2}+ I_{1,0}*K_{1,0} + I_{1,1}*K_{1,1} + I_{1,2} * K_{1,2} + I_{2,0}*K_{2,0} + I_{2,1} *K_{2,1} + I_{2,2}*K_{2,2}) \quad (3)$$

$$O_{1,1}=(10*(1/9)+14*(1/9)+254*(1/9)+ 11*(1/9) + 248*(1/9)+254*(1/9)+ 254*(1/9)+255*(1/9)+255*(1/9) \quad (4)$$

$$O_{1,1} = (10 + 14 + 254 + 11 + 248 + 254 + 254 + 255 + 255)/9 = 172.77 \quad (5)$$



Doing this for all pixels, the following matrix O is obtained:

O =

31	88	116	94	37
88	173	201	179	94
115	200	253	207	123
91	176	205	179	94
34	26	91	94	37

Then, in general:

$$O_{i,j} = (I_{i-1,i-1} * K_{-1,-1} + I_{i,j-1}*K_{0,-1} + I_{i+1,j-1}*K_{1,-1}+ Ii-1,j*K-1,0 + Ii,j*K0,0 + Ii+1,j * K1,0 + I_{i-1,j+1}*K_{-1,1} + I_{i,j+1} *K_{0,1} + I_{i+1,j+1}*K_{1,2}) \quad (6)$$

In the case of a 3x3 grid:

$$O_{i,j} = \sum_{n=-1}^{n=1} \sum_{m=-1}^{m=1} I_{i+m,j+n} K_{m,n} \quad (7)$$

For the case of a 5 x 5 grid:

$$O_{i,j} = \sum_{n=-2}^{n=2} \sum_{m=-2}^{m=2} I_{i+m,j+n} K_{m,n} \quad (8)$$

Where *i* and *j* are indexes of the matrix image, m and n are the indexes of the grid of sampling area, called the core. As a result of the filter implementation, a minor variation between the colors of pixels is obtained, and the noise causing elements are deleted. Figure 1 shows a 3x3 and 9x9 grid.

a) b) c)

Figure 1. Mean Filter window application: a) original image, b) 3x3 Mean filter and c) 9x9 Mean filter

3. RGB Filter

Once a smooth image is obtained, in which some noise generating elements have been removed, then an RGB image filter is used. This filter focuses on RGB primary colors. Depending on the selected color, all the different colors are reduced: If red is selected for each pixel, then:

$$R = (R-B) + (R-G) \quad (9)$$

$$G = 0; \quad (10)$$

$$B = 0; \quad (11)$$

R is normalized referred to the value of red.

$$R = (R/R_{max}) * 255 \quad (12)$$

Based on the above formula, white pixels get a value at zero, thus removing white light, while the pure primary colors (R = 255, B = 0, G = 0), R doubles their value.

Due to the standardization, dark pixels can increase its intensity and generate noise in the resulting image. That is why a minimum value is determined, below which the pixel are considered black (0).

In this case, the selected value is 100. Prior to normalize and calculate the new RGB values, the selected value can also be a value that removes the pixels that are not red or not red enough. This element is called Hue and so the selected value is set in 50. The Hue value is calculated by:

$$H = \cos^{-1}\left[\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right] \quad (13)$$

$$\text{If,} \quad B > G, \quad H = 2\pi - H \quad (14)$$

which gives a range of 0 to 1 and later transforms into a range of [0 - 255]. Once all pixels are obtained, the values under 50 are filtered.

When the filtering process is finished, an image in RGB scale is built with the calculated values in the red component. Figure 2 shows the result.

Figure 2. a) original image, b) RGB filter applied

4. Mass Center and Gravity Center

After the desired pixels are obtained, it is possible to calculate the mass center of the set of pixels. In a discrete system, the center of mass is the geometric point in which the resultant force of all external forces is applied. The center of mass is defined by:

$$mc = \frac{\sum m_i r_i}{\sum m_i} \quad (15)$$

where *m* is the mass of the particle, and *r* is the vector position of a particle. This coincides with the gravity force if the gravitational field is uniform. It has the same magnitude and the same direction at any given point.

If the above formula is associated with each pixel, the intensity *I* is calculated for each pixel:

$$I = (R + G + B) / 3 \quad (16)$$

If every *m* is substituted by *I*, then the center of mass or gravity is:

$$mc = \frac{\sum I_i r_i}{\sum I_i} \quad (17)$$

Where *r* is the point (x,y), for each pixel. It is possible to calculate the *x* component and *y* to the center of mass or gravity, decomposing *r* in the components *x* and *y*.

$$\overline{x} = \frac{\sum I_i x_i}{\sum I_i} \quad (18)$$

$$\overline{y} = \frac{\sum I_i y_i}{\sum I_i} \quad (19)$$

Having obtained the center of gravity or mass, it is possible to know the position of the set of pixels related to coordinate system of the image, thus allowing the robot motion. For example, if the next set of pixels in gray scale is used, with measures of intensity, the center of gravity is:

I =

10	14	254	23	11
11	248	254	253	50
254	255	255	254	255
13	252	254	251	41
13	32	253	35	10

$$\sum I_i = 3555$$

$$\sum I_i * x_i = 7257$$

$$\sum I_i * y_i = 7167$$

Then, the gravity centre is:

$$\bar{x} = \frac{7257}{3555} = 2$$

$$\bar{y} = \frac{7167}{3555} = 2$$

5. Application

To locate the position of the object within the image, the total area of the image can be divided in three regions. The width of the image is split so that the first region is in the left hand side, the second in the center and the third to the right. How to split the image depends on the approach: depending on the region where the center of mass of our object is, the robot turn to the left or right or stop.

In this way, if the center of mass of the red stain is located in the left region, the robot will turn left, and if the center of mass is located to the right turn, the robot will turn right and if it is in the center, the robot will remain static.

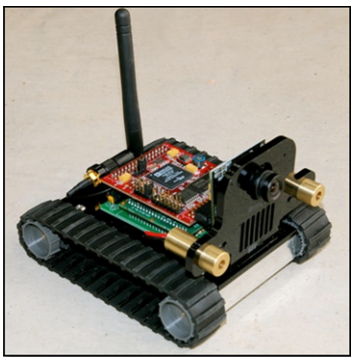


Figure 3. Experimental Robot

The robot used is Surveyor SRV-1 Blackfin Robot Open Source Wireless Mobile Robot with Video for Telepresence, Autonomous and Swarm Operation. The robot is shown in Figure 3.

The image in Figure 4 shows the way in which the image is divided. When "x" is less than 3/8 width, it is considered the left region. When “x” is greater than 5/8 of the image width it is considered the right region. The Center will be located between these two regions.

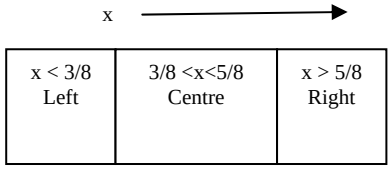


Figure 4. Image Regions, "x" increases to the right

On the other hand if the number of pixels is different from zero, the stain area of the selected color can be estimated. With this area it is possible to get the depth of the object: if the object moves away, the image area decreases, and if the object is approaching to the camera’s field of view, then the stain area of the color increases.

This area within the field of view is defined as

$$A = \sum px_i \qquad (20)$$

where px_i is the i_{th} pixel,

if $I > 0$ then: $px = 1$,

otherwise: $px = 0$

When a range of areas is selected to stop the robot and is placed at a suitable distance, the two levels are taken as the upper and lower limits. If the value of area is greater than the upper limit, the robot will move backwards and if, on the other hand, it is less than the lower limit, the robot will move forward. So when the robot is close to the ball, it will move backward and if the ball is far it will move forward. The robot stops when it has an area that is within the pre-defined range.

“x” is a component of the center of mass of the object. The area of the red stain is the area of the object. This process is made for each captured image. If no values are obtained or there is no red object, the robot begins to search slow, turning 360 degrees. A lower limit of 2025 pixels and an upper 7725 pixels has been established in the area of the object. These

values are set according to the circumstances and they can be changed.

6. Evaluations and Results

Table 1 shows the ball area obtained by robot SRV1-blackfin video camera, at different resolution. The distance of the robot and the ball is 1 meter. Also, it shows the recognition time. For an image of resolution 640x480 pixels, it takes 344.82 to 434.82 milliseconds, depending on the environmental conditions.

Table 1 Recognition time and image resolution

Image Resolution (jpg)	Object Area	Recognition time (milliseconds)
160 x 120	100 pixels	48.076 – 48.78
320 x 240	324 pixels	131.57 - 156.25
640 x 480	784 pixels	344.82 – 434.82

It is observed that with a smaller image, the recognition is faster. The time is measured from the robot to the red ball, to a distance of one meter between them. In this time, the center of gravity and the size of the ball is determined by a program. The area has to be well illuminated since the lighting conditions have a key role in the recognition task.

The result of the implementation after processing the image is that the robot follows red objects, i.e., the ball. Figure 5 shows the original image (a). Image (b) shows the filtered image using the Mean Filter. Image (c) shows the red or close to red filter image with the RGB filter. Finally, the center of the object is in image (d).

Finally, this information is processed to obtain the mass center values and the area of the object allowing the movement of the robot.

a) Original image b) using mean Filter

c) with RGB filter d) center of the object

Figure 5. Image processing results

5. Conclusions

This paper has presented the integration of several image processing algorithms necessary to recognize a particular color and the movement of an object. The main objective is to detect the object by its color and track it by a mobile robot. Mean filter is applied to soften and sharpen the input image. Then RGB filter is applied to calculate the center of mass and area of the object and to locate its position in a real environment.

It is a basic implementation for mobile robot vision, which is a colored object recognition in motion.

Perhaps a drawback of this approach is the noise sensitivity since if there are multiple objects within the same color application, it will not function properly. It works very well in low noise environments and with one particular object. Therefore, it is important to have a controlled environment in which this noise is avoided.

The image refreshing speed is another issue, because the robot movements are fast, but image capture is slow, Frames Per Second (FPS) rate is very low (varies from 1 to 3) and sometimes there are regions where the image is lost or is not captured. In this case, the system returns to find the image again. For this reason, the system takes pictures of 160 x 120 pixels, decreasing the speed of the servo. An alternative solution is to eliminate image refreshing, thus increasing the transmission bandwidth and image acquisition.

With the integration of these algorithms for image processing and the increase of image acquisition bandwidth, the object's color recognition was achieved, practically in real time, which makes it very useful for applications of mobile robotics or in manufacturing tasks.

REFERENCES

1. OSORIO, R., M. PEÑA, G. ACOSTA, C. TORRES, **Clasificador de Color utilizando Procesamiento de Imágenes**. X Congreso de la Asociación Chilena de Control Automático, 1992. pp. 51-56.
2. OSORIO, R., G. AGUILAR, **Reconocimiento de Trayectoria para un Robot Móvil Aplicando Teoría del Color,**

XII Congreso Chileno de Ingeniería Eléctrica, Temuco, Chile, 1997.

3. OSORIO, R., F. PESSANA, **Cuantificación y Reconocimiento de Formas Utilizando procesamiento de Imágenes,** International Conference on Automatic Control PADI2, Piura-Perú, 1998. pp. 76-82.
4. OSORIO, R., F. PESSANA, M. PEÑA, J. SAVAGE, **Reconocimiento de Objetos Aplicados a un Sistema de Visión para Robots** World Multiconference on Systemics Cybernetics and Informatics. Orlando, Florida, USA, 1998. pp. 113-117.
5. OSORIO, R., M. PEÑA, C. SAN MARTIN, **High Dynamic Range Analysis Method for Color Image Enhancement**, Jornadas chilenas de computación, 2009. pp. 224-228.
6. LEE, K. E., W. CHOE, J.-H. KWON, S. LEE. **Locally Adaptive High Dynamic Range Image Reproduction Inspired by Human Visual System**, Proc. SPIE 7241, 72410T (2009); doi:10.1117/12.806057.
7. GONZALEZ, WOODS, **Digital Image Processing**. ISBN 9780131687288, Prentice Hall, 2008.
8. FISHER, R., S. PERKINS, A. WALKER, E. WOLFART, **Hypermedia Image Processing Reference**, Published by J. Wiley & Sons, Ltd. 2003.
9. BUGEAU, A., P. PEREZ, **Detection and Segmentation of Moving Object in Highly Dynamic Scenes**. IEEE Conference on Computer Vision and Pattern Recognition CVPR '07, 2007. pp. 1-8.
10. SCHLEYER, G., G. LEFRANC, **Tridimensional Visual Servoing**. Studies on Informatics and Control, Vol 18. No. 3, 2009. pp. 271-278.
11. VERNON, D., **Machine Vision**, Chapter 4, Prentice-Hall, 1991.
12. GEVERS, TH. A. W. M. SMEULDERS, **Image Indexing using Composite Color and Shape Invariant Features**, Int. Conf on Computer Vision, Bombay, India, 1998. pp. 234-238
13. SWAIN, M. J., D. H. BALLARD, **Color Indexing**, International Journal of Computer Vision, Vol. 7, No. 1, 1991. pp. 11-32.
14. FUNT, B. V., G. D. FINLAYSON, **Color Constant Color Indexing**, IEEE PAMI, 17(5), 1995, pp. 522-529.
15. LIPTON, A. J., H. FUJIYOSHI, R. S. PATIL, **Moving Target Classification and Tracking.** Proceedings of Fourth IEEE Workshop on Applications of Computer Vision WACV '98, 1998. pp 8-14.
16. KIM, S., S. LEE, S. KIM, J. LEE, **Object Tracking of Mobile Robot using Moving Color and Shape Information for the aged walking.** International Journal of Advanced Science and Technology, Vol. 3, 2009, pp 293-297.
17. LIU, F., M. GLEICHER, **Learning Color and Locality Cues for Moving Object Detection and Segmentation.** Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 320-327.
18. FANG, X. H., W. XIONG, B. J. HU, L. T. WANG, **A Moving Object Detection Algorithm Based on Color Information**, Journal of Physics: Conference Series Vol. 48, 2006, pp 384-387.
19. DONG L., L. XI, **Monocular-vision-based Study on Moving Object Detection and Tracking,** 4th International Conference on New Trends in Information Science and Service Science (NISS), 2010, pp. 24-29.
20. KARLSSON, N., E. DI BERNARDO, J. OSTROWSKI, L. GONCALVES, P. PIRJANIAN, M. E., MUNICH, **The vSLAM Algorithm for Robust Localization and Mapping.** IEEE International, Conference on Robotics and Automation, Proceedings, 2005. pp. 24–29,
21. LOWE, D. G., **Object Recognition from Local Scale-Invariant Features.** Proceedings of the Seventh IEEE International Conference on Computer Vision, 2, 1999, pp. 1150-1157.
22. BROSNAN, T. D.-W. SUN, **Improving Quality Inspection of Food Products by Computer Vision - A Review,** Journal of Food Engineering, v 61, n 1, 2004, pp. 3-16.

23. TOPOLESKI, L, D., **Hort 220 Vegetable Identification,** Yard & Garden Line News, Volume 2 Number 6 May 1, 2000.

24. ZHOU, L., V. CHALANA, Y. KIM, **PC-based Machine Vision System for Real-Time Computer-Aided Potato Inspection,** International Journal of Imaging Systems and Technology, v 9, n 6, 1998, pp. 423-433.

25. NOORDAM, J. C., G. W. OTTEN, A. J. M. TIMMERMANS, B. H. VAN ZWOL, **High Speed Potato Grading and Quality Inspection Based on a Color Vision System**, Proc. SPIE Vol. 3966, Machine Vision Applications in Industrial Inspection VIII, Kenneth W. Tobin; (Editors). 2000, pp. 206-217.

26. PEÑA-CABRERA M., I. LOPEZ-JUAREZ, R. RIOS-CABRERA, J. CORONA-CASTUERA, **Machine Vision Approach for Robotic Assembly**, Journal of Assembly Automation, ISSN 0144-5154, Vol. 25, No 3, 2005. pp 204-216.

27. LEIGHTON F., R. OSORIO, G. LEFRANC, **Modeling, Implementation and Application of a Flexible Manufacturing Cell,** International Journal of Computers Communications & Control, ISSN 1841-9836, 6(2). 2011. pp. 278-285.