ROBOTICS AND VISION RECOGNITION FOR INTRODUCTION TO FLEXIBLE AUTOMATION

Ph.D. Blagoja Samakoski, Ph.D. Aleksandar Markoski, M.Sc. Zlatko Sokolski,
M.Sc. Aleksandar Bučkovski, B.Sc. Aleksandar Miloševski, B.Sc. Aleksandar Sekuloski
MIKROSAM A.D. "Krusevki pat" b.b. tel. +389 48 400 100 info@mikrosam.com.mk

Abstract

Vision and tactile sensing capability are gradually being incorporated in robotic work-cells in the modern factories as part of flexible production systems, in order to achieve additional flexibility to parts handling in pick and place, welding, painting, polishing and assembling operations. In this project we used a robotic arm and made computer program to demonstrate the process of flexible control in robotic work-cell for locating, recognition, pick and place operations.

Key words: Robotics, Vision recognition, Flexible automation.

1. PROLOGUE

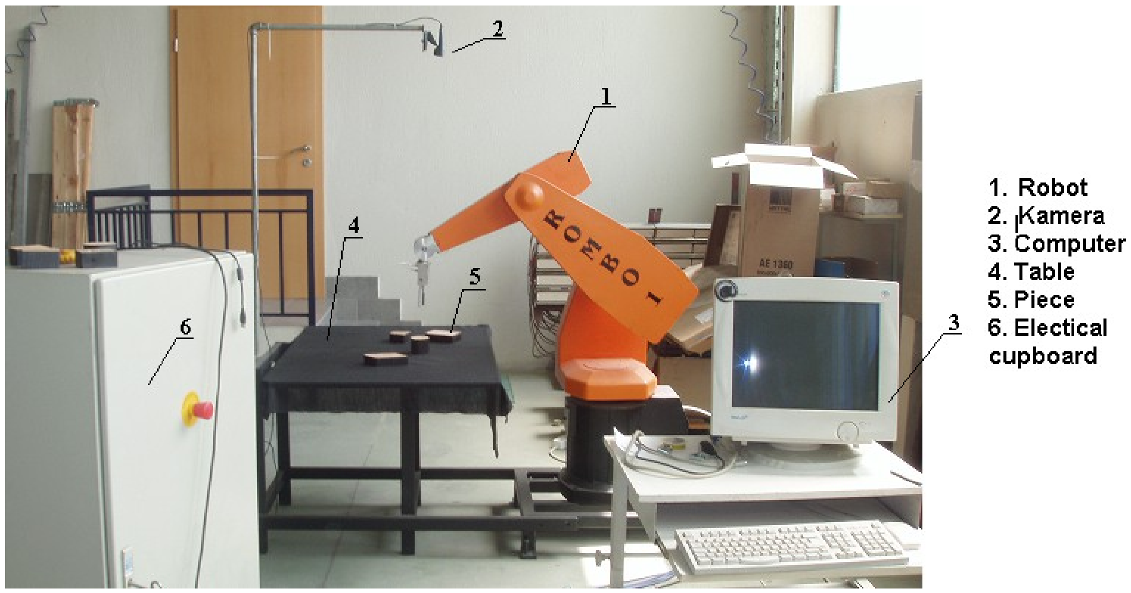
Operation that include pick,place and palletize is done with manipulators as routine in today's factories. Tasks without contact for example welding, painting or removing paint successfully are automated, but attempts for cleaning and polishing with robot help as well as assembling of complex parts have leaded to new challenges which robots should be exceeded in the future. Torsion sensors and other sensetivity sensors give robots opportunity to fulfill tasks with contact.

Besides force sensors and sensitivity, efforts for industry automation long ago recognized benefits of integration of visual recognition for flexibility in working cells of the robots.

In this study will be demonstrated better flexibility gained with integration of visual recognition with robot's working cell. This project is still in developing phase and it is designated demonstrating concepts of flexible automatisaton. Main objective of this project is different form visual recognition, finding its position at the table, and moving it to designated place. This is simulation of a working cell in which robot would have task to transport parts from motion track to palette.

2. EXPERIMENTAL EQUPMENT

Experimental equipment for the above described task contains: PUMA robot equipped with pneumatic griper for catching elements, web kamera Kodak fixed above the working table, electric cupboard and computer with two experimental softwares Eggbracker (for robot controlling) and VFW (for imageprocesing) Computer and robot have communication through Gallil motion controller placed in cupboard. Pic. 1 show experimental equipment which was used in this project. Robot's working area which is in camera's focus it's covered with black fabric to increase contrast between pieces and table. Pieces are made from wood in many different forms.



Pic 1

3. ROBOT KINEMATICS

Robot controlling and programming need kinematics knowlidge. For that purpose analytic study of manipulator's motion geometry has been done compare to referent coordinate system, same time taking calculation of forces or moments that cause that kind of moving. Between referent coordinate system and coordinate system of piece kinematics we have dependency which basically means translation and rotation.

The use of kinematics describing space configuration of robot, allows two approaches :

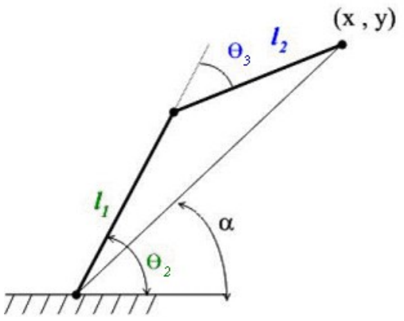
- **Direct kinematic**-angles of juncture of robot are given, orientation and position of griper need to be found
- **Inverse kinematics** - orientation and position of griper are given, angles of juncture need to be found

3.1. Inverse Kinematics

There are two methods for solving this problem, algebraic and geometric. In this study we have used geometric method for angle of rotation calculation for every juncture separately. Considering referent coordinate system, rotation of first axis depends only of x and y cooridinates of final griper position, as follows:

$$\theta_1 = arctan\left(\frac{y}{x}\right).$$

Little bit more complex task is calculation of θ_2 and θ_3 angles. Using geometric methode (see Pic2):



$$c^2 = a^2 + b^2 - 2ab\cos C; \quad \left(x^2 + y^2\right) = l_1^2 + l_2^2 - 2l_1 l_2 \cos(180 - \theta_3)$$

$$\cos(180 - \theta_3) = -\cos(\theta_3);$$

$$\cos(\theta_3) = \frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2}; \theta_3 = \arccos\left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2}\right)$$

$$\frac{\sin B}{b} = \frac{\sin C}{c}; \quad \frac{\sin \bar{\theta}_1}{l_2} = \frac{\sin(180 - \theta_3)}{\sqrt{x^2 + y^2}} = \frac{\sin(\theta_3)}{\sqrt{x^2 + y^2}}; \quad \theta_2 = \bar{\theta}_2 + \alpha;$$

$$\alpha = \arctan 2\left(\frac{y}{x}\right); \quad \theta_2 = \arcsin\left(\frac{l_2 \sin(\theta_3)}{\sqrt{x^2 + y^2}}\right) + \arctan 2\left(\frac{y}{x}\right).$$

Pic 2

Angle θ_4 always tries to hold vertical position down to griper, while angle θ_5 depends of rotation of piece and current rotation of first axis.

$$\theta_4 = \alpha \quad u \quad \theta_5 = \delta - \theta_1 .$$

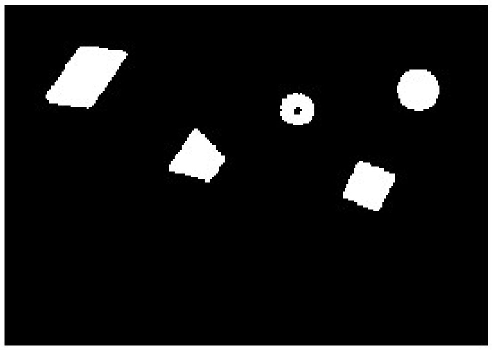
4. VISUAL DATA PROCESSING

- *Image processing phases;*
- *State saving;*
- *Gaining black-white image with help of the histogram;*
- *Gaining pieces outlines (convolution);*
- *Extracting contours from outlines;*
- *Group contours depending of piece affiliation;*
- *Center calculation (centroid);*
- *Rotation angle calculation ("Hough" transformation);*
- *Form recognition;*

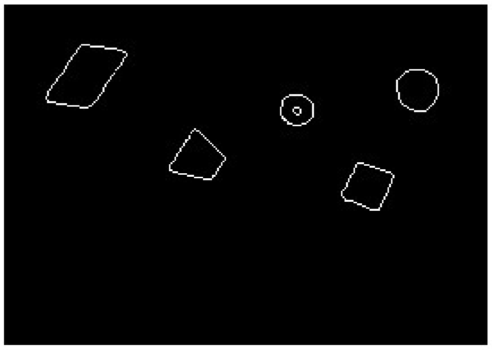
4.1. Gaining black-white image

To process and analyze image of robot's working field, first we have to "catch" (save) current state. This way we obtain statically color image (Pic 3a). Next step is converting image into black-white. This mostly has been made by making a histogram. (Pic 3g). The Histogram shows number of pixels with same color intensity found on the image. To obtain only black and white pixels, threshold (border) value has been set. All pixels under that value will be black and above that value will be white. This threshold (border) value depends of values obtained with the histogram. Result of this operation is Pic 3b).

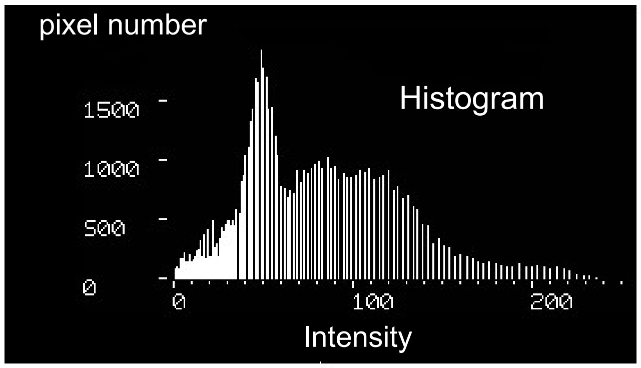
a)



b)



c)



d)

Pic 3

4.2. Gaining pieces outlines (convolution)

In further image analyse we need to obtain outlines which include discrete form of 2D spiral (convolution) operator. It defines itself with next relation between pixels $g_i\ (x, y)$ of input image, pixels $h(\varepsilon,\ \eta)$ of spiral core **H** and pixels $g_0\ (x, y)$ of output image [1]:

$$g_0(x,y) = \sum_{\varepsilon=-\infty}^{\infty} \sum_{\eta=-\infty}^{\infty} g_i(\varepsilon,\eta) \cdot h(x-\varepsilon, y-\eta),$$

where *x, y,* ε and η are integer numbers. Signtifical callculated advantages can be achived by introducing difrent cores $h(x-\varepsilon, y-\eta)=h(x-\varepsilon)h(y-\eta)$. Difrent cores give opportunity of changing 2D operation with two successful 1D operations. Spiral operator in our case is:

$$H(u,v)_{NNA} = \frac{1}{9}\begin{bmatrix} -5 & -5 & -5 \\ -5 & 39 & -5 \\ -5 & -5 & -5 \end{bmatrix}$$

Result of this way processed image has been shown on Pic 3c .

4.3. Extracting contours from outlines

After converting image to black and white we need to find contours of each object. Algorithm for that is [2]:

1. Searching image line by line starting from up left corner. When white pixel is found, its coordinates are transferred to array and its color is changed in black;
2. Searching is based on color analyze of 8 pixels around last pixel putted in array:
    - If white pixel has been found (in 8 pixel surroundings), coordinates of this pixel are put at the end of the array and its color is changing in black. Jump to step2.
    - If white pixel hasn't been found, array has been put to list of not connected (open) outlines and algorithm is returning to step 1.

Describe algorithm is simple but effective. After point is added in array, direction in which last white pixel been found has been saved. In searching for other pixels, priority is given to pixel closest to saved direction.

After this phase, all open outlines should be closed. Contours containing one pixel are been thrown away. Before going to connecting phase, it should defined:

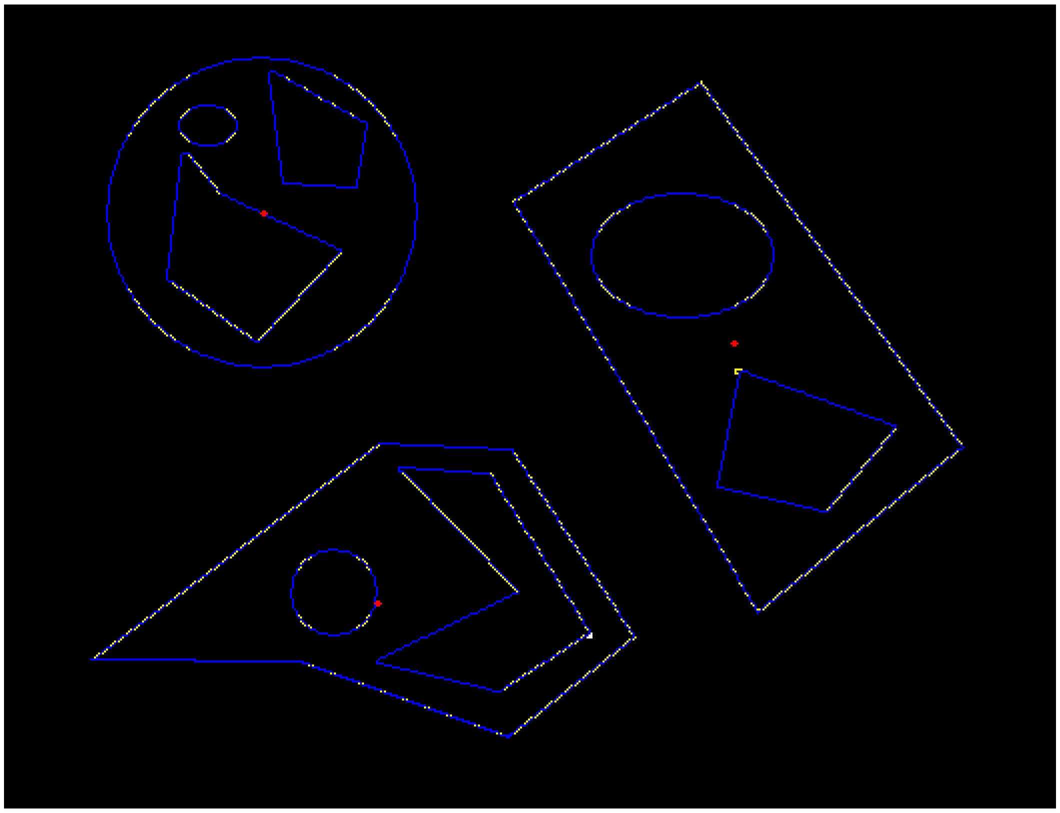
- Minimal size (number of pixels) of open outline. Contours shorter then 2 pixel have been erased;
- Maximal distance (in pixel) between end points of open outlines that will connect;

Contours are been connected in the following way:

1. Adjusting minimal distance to 2 pixels;
2. Comparing ending points of outlines. If distance between ending points of any two outlines is lower or equal then 2, then we are adding missing pixel and connect outlines together. If we discover that open outline has been closed, we add it to list of closed outlines (contours);
3. Step 2 is repeating itself, while there are outlines that can be connected;
4. When there are no more combinations, algorithm ends;
5. At the end, erase all not closed outlines;

4.4. Group contours depending of piece affiliation

Using PointInPolygon algorithm we group contours depending of piece affiliation as you can see on Pic 4 (only 3 objects are found).



Pic 4.

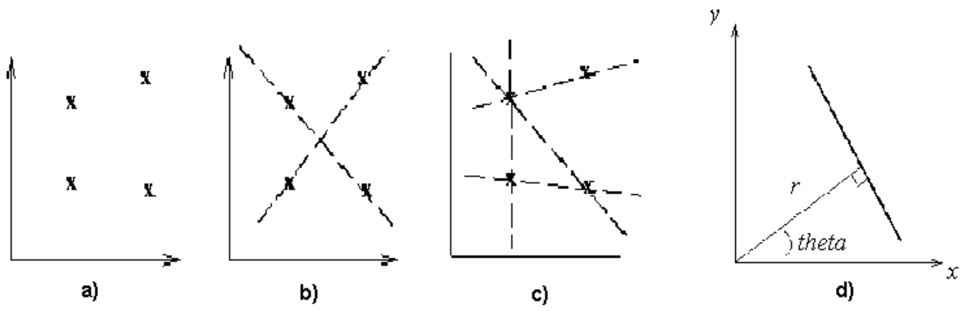
4.5. Center calculation (centroid)

Next step defines centroid by using this formulas (red dot on Pic 4):

$$X = \sum_{i=1}^{n} x_i;\ Y = \sum_{i=1}^{n} y_i;\ x_c = \frac{X}{n};\ \ y_c = \frac{Y}{n}.$$

4.6. Rotation angle calculation ("Hough" трансформација)

"Hough" transformation [3], is technic with its help we can define orientation direction of some form in the image. Main idea in line detection process is that every dot of the form have influence on a global solution. For example we can take one problem of finding dots direction shown. Pic 5a). Possible solution are shown at Pic 5b) and c).



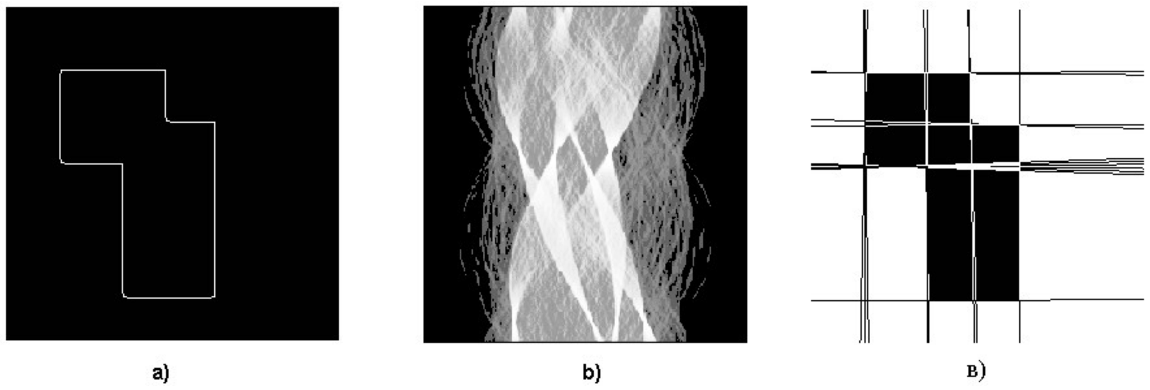
Слика 5

Linear segment analytical can be presented on different ways. But, for our case best chose is equation:

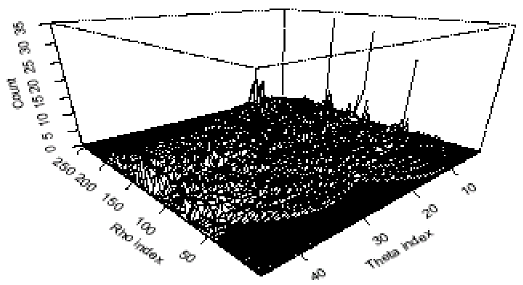
$$x \cdot cos\theta + y \cdot sin\theta \ = r,$$

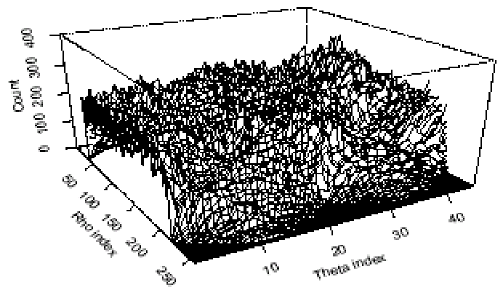
where r is normala from center to the line and θ is angle of obeisance (Pic 5d)).

Form points (x_i, y_i) on image are known and used for parametric equation, while r and θ are unknown values which are being searched for (Pic 6 a)). If we draw all posible $(0<r<r_{max}, 0<\theta<360^o)$ values defeint by all (x_i, y_i), points into polar coordinate system they will be projected as curves (sinusoids Pic 5 b)). This point-to-curve transformation present "Hough" transformation of lines. If we look in a polar coordinate system, the points which are collinear in Dekart's coordninate sytem, in polar they form curves intesecting each other. Transformation is implementing coordinate system on finite intervals or accumulators cells ($A_{r\, x\theta\, x\, counts}$ multidimensional field) . As algorithm works of, every point (x_i, y_i) is transforming itself to discrete (r, θ) curve, by that every point (vote) increase the counter value in the accumulators cell (Pic 7). Local maximus (point where the most sinusoids intersecting each other) in accumulator's cells correspond to lines defining object orientation direction (Pic 6c)).



Pic 6





Pic 7

4.7. Form recognition

Since perceptual structure (form) is basic property of objects we need effective method of description. There are several methods for object form description and recognition. (like Zeinrike, Fourier descriptors…) , in this study we have used geometric moment descriptors.

Central moments are *p+q* order for 2D form given by function *f(x,y)* has been given with next equation:

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q f(x,y) \quad p,q = 0,1,2,... \quad \bar{x} = \frac{\mu_{10}}{m},\ \bar{y} = \frac{\mu_{01}}{m}$$

where *m* is the weight of form region, μ_{pq} are translation constants . First seven normalized geometrical moments which are unchangeble to translation, rotation or skaling, has been given with next equation:

$$\Phi_1 = \eta_{20} + \eta_{02}$$
$$\Phi_2 = (\eta_{20} + \eta_{02})^2 + 4\cdot(\eta_{11})^2$$
$$\Phi_3 = (\eta_{30} - 3\cdot\eta_{12})^2 + (3\cdot\eta_{21} - \eta_{03})^2$$
$$\Phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$\Phi_5 = (\eta_{30} - 3\cdot\eta_{12})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3\cdot(\eta_{21} + \eta_{03})^2\right] + (3\cdot\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})\left[3\cdot(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]$$
$$\Phi_6 = (\eta_{20} - \eta_{02})\left[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right] + 4\cdot\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$\Phi_7 = (3\cdot\eta_{21} - \eta_{30})(\eta_{30} + \eta_{12})\left[(\eta_{30} + \eta_{12})^2 - 3\cdot(\eta_{21} + \eta_{03})^2\right] + (3\cdot\eta_{12} - \eta_{03})(\eta_{21} + \eta_{03})\left[3\cdot(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2\right]$$
$$\eta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^\gamma} \qquad \gamma = 1 + \frac{p+q}{2} \qquad p+q = 2,3,...$$

In recognition process first we form database with moments of all known objects. Then we compare calculated moments with moments in database. This comparation is done using next formula:

$$\delta = \sum_{i=1}^{7} (\Phi_i - \Phi_{ib})$$

where Φ_i are calculated moments, while Φ_{ib} are moments from database. If δ is lower then 0,001 we have recognized some piece.

5. CONCLUSION

Implementation of a system of visual recognition is complex problem. The biggest problem is correct processing of visual data. Only a correct processing will lead to a correct robot working. Algorithms described in this study rich some degree of processing of visual data.

In other way this study is illustrative example how robot working cells can increase its flexibility in operation of choosing, putting and palletizing. In modern factories, flexibility obtained with integration of system for visual recognition give opportunity of eliminating expensive, difficulties and specific devices for manipulation.

6. LITERATURE

[1] Michael Seul, Lawrence O'Gorman, Michael J. Sammon; "Practical Algorithms for Image Analysis" USA 2000;
[2] Jiri Stastny; "Image Processing by Using Wavelet Transform" Technical University of Borno;
[3] Klaus Hansen, Jens Damgaard Andersen; "Understanding the Hough transform" Department of Computer Science University of Copenhagen Denmark;
[4] Владимир Дуковски; "Роботика" Скопје 1994.