

## VISION-BASED ROBOT PROGRAMMING

**Gabor Sziebig**

Narvik University College, Narvik, Norway

**Peter Zany**

Budapest University of Technology and Economics, Budapest, Hungary

**Abstract:** *This paper presents a new vision based programming methodology which combines information of the real and ideal world, especially adapted for robot grinding and deburring operations. Further, the presented methodology is especially developed with simplicity in mind when it comes to man- machine communication. Thus, a standard marker pen can be used by the operator to draw the robot path directly onto the work piece. This line is captured by a vision system and finally leads to the generation of the robot path.*

**Keywords:** *industrial robot programming, vision system, path planning*

### 1. INTRODUCTION

Many manufacturing processes leave irregularities (burrs) on the surface of a work piece. Burrs are often triangular in shape and is found after casting, forging, welding, and shearing of sheet metals [1]. These irregularities are often identified and removed by human operators using by manual grinding. Manual grinding means hard and monotonous work and the workers need to protect themselves by wearing protecting equipment such as goggles, gloves and earmuffs. [5].

Grinding is mostly carried out at an ending stage in the manufacturing process where the work piece gets its final geometry. Any manufacturing error in this stage could be very costly and even lead to dismissal of the entire work piece. Not accounting for possible operator errors the deburring process itself is said to add up to 10% of the total manufacturing cost [1].

Due to the health risk associated with grinding and the possible added cost from operator errors there is a strong incentive to explore new automated solutions. The industrial robot is viewed as a strong component for this job.

Programming of industrial robots is traditionally done by the *Teach* or the *Offline* programming methodologies. However both methods encounter problems in grinding/deburring operations. The *Teach* method is time-consuming since it requires the operator to manually define all poses (position and orientation) of the tool- centre- point (TCP) on the work piece. Also, the “Teach” methodology carry no information on the ideal work piece, or say the expected end result after the grinding process. In traditional *Offline* programming the robot path is generated from a CAD model of the work piece. This CAD model holds no information on the irregularities (burrs) and then the necessary path cannot be created.

Thus, there is a need to develop new methodologies for programming of industrial robots, especially associated to manufacturing operations like grinding/deburring.

Typically, manual grinding operations is carried out based on individual observations on each work piece. The human operator is very well able to identify the location of these “problem areas”, but cannot clearly state what is the necessary cutting depth and how much material is to be removed to reach the ideal final geometry.

In this paper a modified and improved methodology for vision based robot programming is presented, especially adapted for grinding/deburring operations.

This methodology:

- allows the user to draw the grinding path directly onto the work piece
- uses only one camera system to catch the operator generated path
- calculates automatically the cutting depth (material removal)
- calculates automatically the robot grinding/machining 6- dimensional path

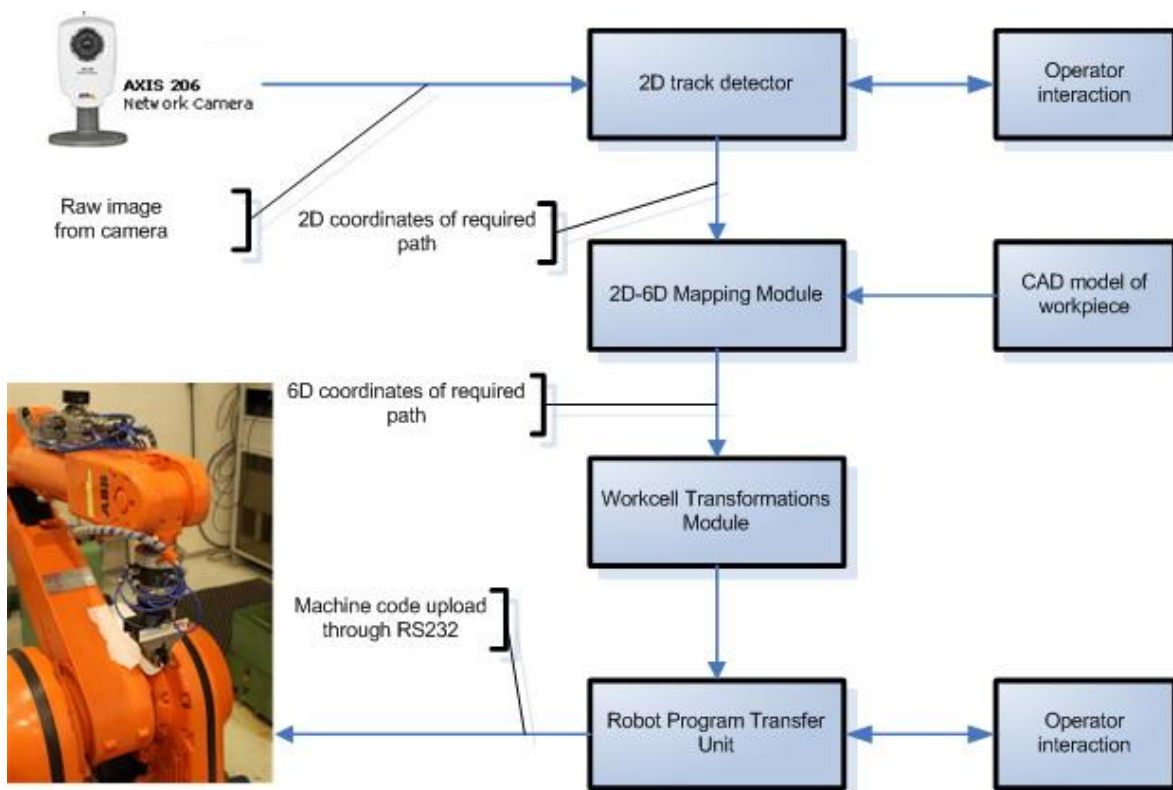
By purpose the methodology is not 100% automatic. The expertise of the worker is still needed and used to identify the locations of areas to be modified. However, the man- machine communication is greatly simplified since the programming is done by moving a standard marker pen.

The organization of the paper is as follows: Section II gives an overview of the programming methodology while section III presents details of system components. Section IV presents experimental results while section V concludes the paper.

## 2. SYSTEM OVERVIEW

Fig.1. summarizes the proposed methodology. According to Fig.1. the image of the real work piece is captured and transferred to the 2D track generator. In this module several image processing tools are available but especially contour lines are developed. Further, the operator should interact with the image to define a work piece coordinate system and then select/define the robot 2D path. This selection is done via a drawing module which allows the operator to generate curvatures onto the existing picture wherever necessary. The generated robot path could be as simple as a straight line, described by a start and an end point, but it can also be a Bezier curve described by 500 or more points. It is the operator's role to make this decision.

Finally, all paths are stored with reference to the work piece coordinate system.



*Fig.1. System overview*

In the next step the two dimensional path is transferred into six dimensional (6D) robot coordinates. This is done in an offline programming tool where the CAD model of the work piece is introduced. Since the 2D track generator holds no information on the depth coordinate ( $z$ ) this is established using an automatic collision (hit and fallback) procedure. Also, in each cutting point, the surface inclination is automatically found and a cutting point coordinate system is generated. Based on the size and geometry of the machining tool the most effective orientation of this tool with respect to the cutting surface is established. At this point, all of the six coordinates necessary to describe a robot path is stored with reference to the work piece coordinate system.

Next step is to establish the relationship between the robot base coordinate system and the work piece coordinate system, out in the machining area. To describe the work piece position and orientation with respect to robot base coordinates a high accuracy procedure is introduced. When the relationship between these two coordinate systems is found the robot path is transferred into robot base coordinates.

The final step in the procedure is to compile the robot program into machine code and execute.

In section III details of selected system components are given.

### **3. SYSTEM COMPONENTS**

#### **3.1 THE 2D TRACK DETECTOR**

The goal of the application was already presented in the previous section, now the details and the transformation steps (from raw image to 2D work piece coordinates) will be presented.

In Fig.2. the application use-case diagram can be seen. In this picture the transformation steps are clearly identified. The program starts with an initial screen, a typical sample as seen in Fig.3.

After choosing the input image source, the image is shown in the left part of the application, in the Image from camera box. If it is a live video or an IP camera the live video is shown. We can stop the video whenever we want, and process the actual image shown in the Input image box. The processed image is shown in the right part of application in the Processed image box. The process means that the picture is sent through image filters, and the resulting image is shown in the Processed image box. The image processing steps, algorithms and filters are presented later in this section.

If we are satisfied with the image processing, we can step forward with pushing View with the Window button, or we can start the whole process from the beginning to get a better result with tuning some parameters of image processing. A sample screen can be seen in Fig.4.

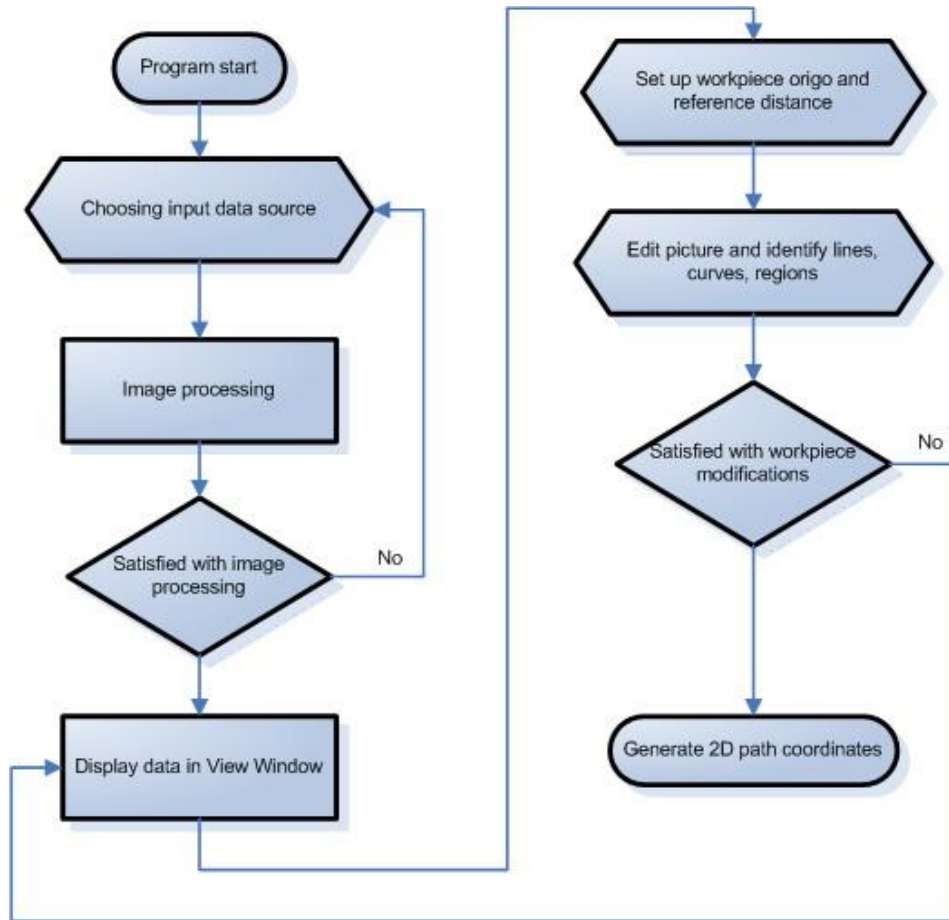


Fig.2. 2D track detector flow chart

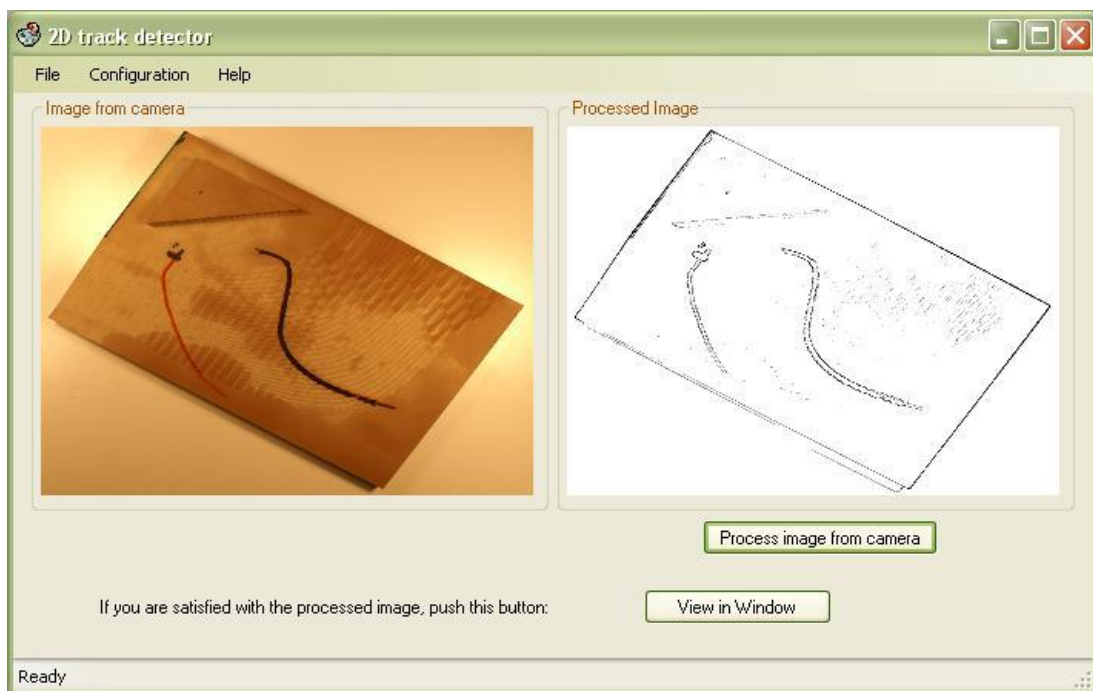
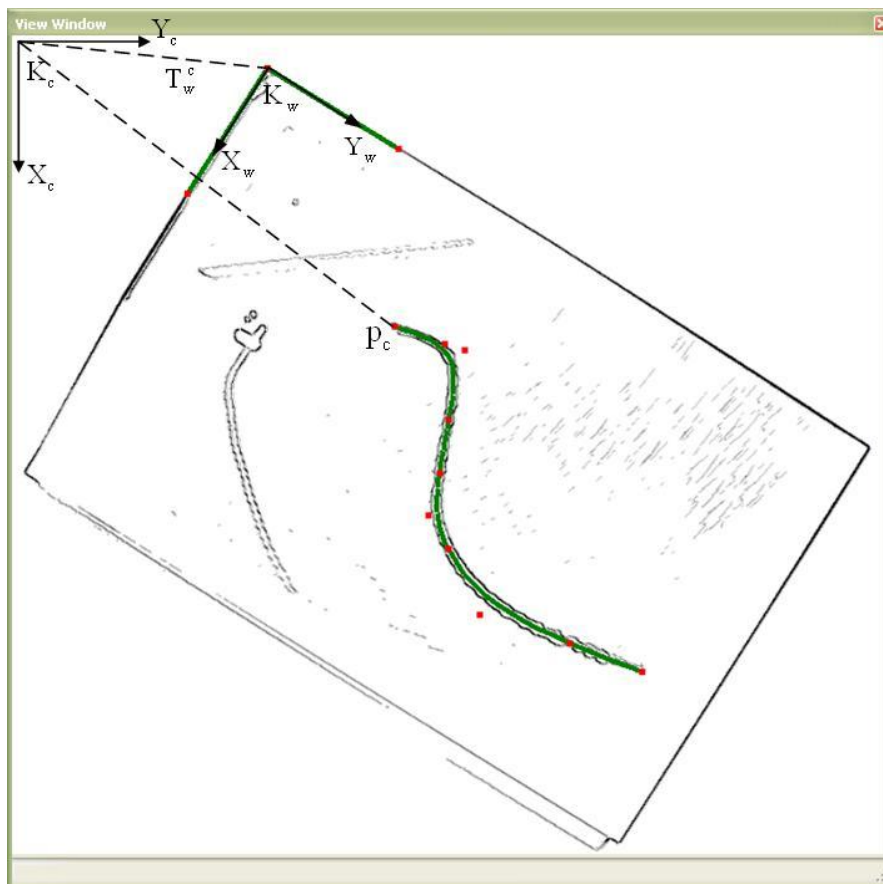


Fig.3. Sample screen of 2D track detector

The View Window shows the same picture as in the Processed image box, but in full size, where the operator can identify the errors of the work piece and can create the robot path by creating lines curves and regions. The line, curve and region functions can be selected from the menu on the bottom of the window. A status box instructs the operator what to do and how many points are needed for one line or curve or region. The following geometrical figures can be created on the surface of the work piece:

- Line: contains one start and one end point
- Curve: contains four points, one start and one end point, and two control points (Represented as a Bezier curve)
- Curves: contains connected curves, connection on end and start point
- Region: contains at least three points



**Fig.4.** Sample screen of View Window

In the View environment the selected grinding/deburring paths are stored as 2D data points with reference to a camera coordinate system ( $K_c$ ). However, to make these points portable they should be transferred to a coordinate system on the work piece, preferable the same coordinate system ( $K_w$ ) used by the CAD system. Prior to set-up of this work piece coordinate system the pixels positions are scaled according to the distance from the camera lens. The work piece coordinate system ( $K_w$ ) is then set-up and defined from a selection of well known basic geometric elements found at the work piece. A unit vector ( $z_w$ ) normal to the picture plane is selected as one of the coordinate axis, while typically, a line on the surface

form basis for the second unit vector direction ( $x_w$ ). The third axis ( $y_w$ ), in this right hand coordinate system, is then given as the cross product between the two other unit vectors.

$$y_w = z_w \times x_w \quad (1)$$

The origin coordinates of  $K_w$  are selected, typically, as the intersection of two non-parallel lines with the  $z$  coordinate set to 0. Pixel position  $p_c$ , originally stored in  $K_c$ , can now be transferred into  $K_w$  coordinate system by the transformation matrix  $T_w^c$ .

$$[p_w \ 1] = [p_c \ 1] T_w^c = [p_c \ 1] \begin{bmatrix} \cos \phi & \sin \phi & 0 & 0 \\ -\sin \phi & \cos \phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x & y & 0 & 1 \end{bmatrix} \quad (2)$$

In (2) the angle  $\phi$  is the angle between  $x_w$  and  $x_c$  axis, while the coordinates  $x$  and  $y$  are the distance between the two coordinate system origins.

The View application does not know anything of the CAD model of the work piece; it is mainly for identifying errors. However it is possible to match the work piece with the original CAD model. The opacity of View Window can be modified from a menu, and the window can be moved over a CAD modeler.

Finally, by pushing the save button, the 2D coordinate path will be stored with reference to the work piece coordinate system.

### 3.1.1 IMAGE PROCESSING [4]

The goal of the image processing, in the 2D track detector, is to reveal the errors. This can be achieved by using a sequence of image filters. The sequence steps can be seen in Fig.5.

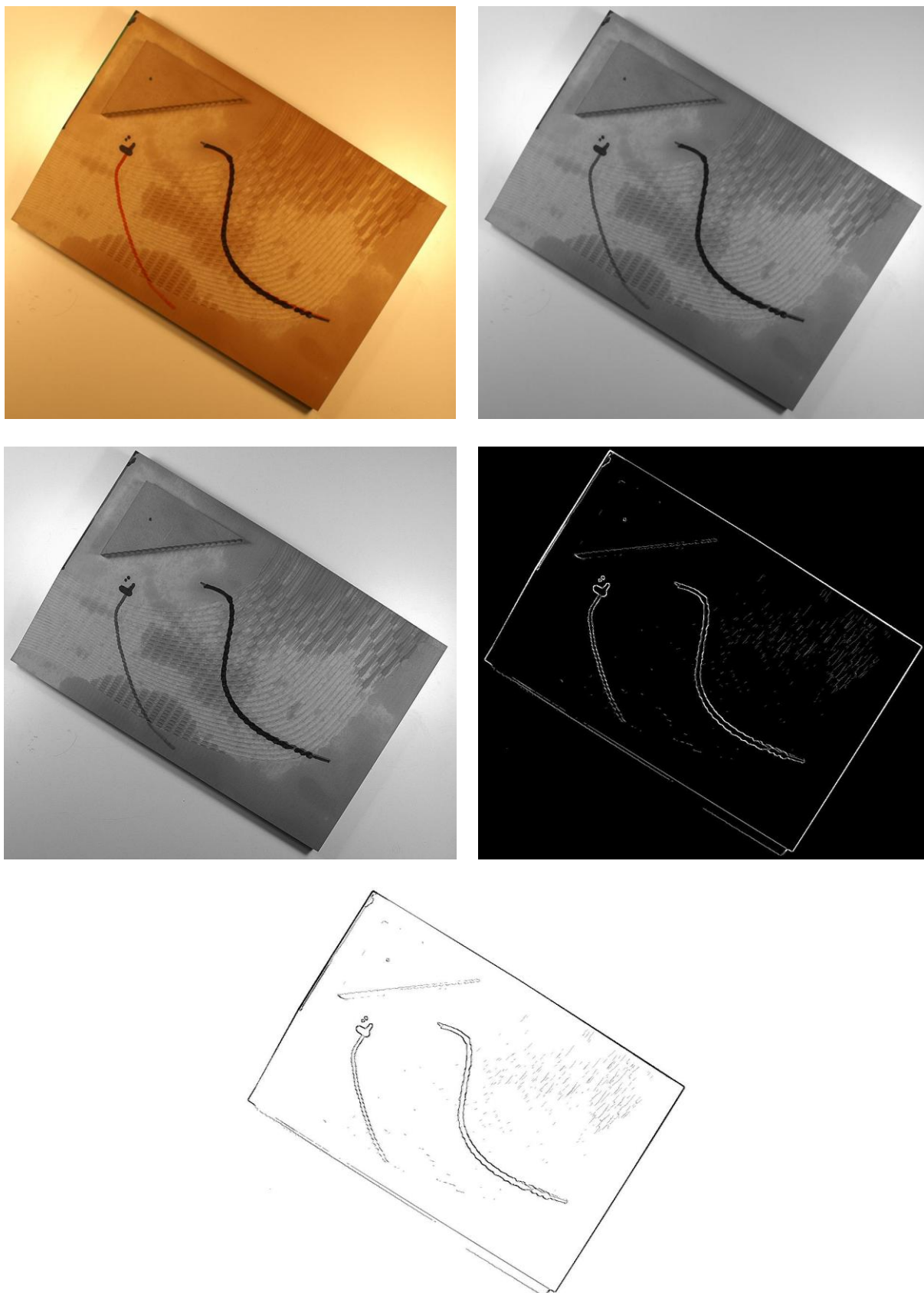
The sequence starts with a grayscale conversion, this step cannot be left out, because it is faster and easier to apply filters to a grayscale image. A grayscale image only contains one byte information per pixel, what is great reduction compared to three byte per pixel.

The next step is the sharpening filter. This is a pre-processing step for the edge detection. Sharpening is also a convolution filter, and can be given with its filter matrix.

With a sharpening filter, the contour of the objects in the image is accentuated. Other type of pre-processing filters could also be applied, but the sharpen filter is the most commonly used.

Next a Canny edge detector is used. There exists much kind of edge detectors, but our experiments show that, this type is the best for error detection.

The last step in our sequence is the inverting. This step could be left out, if the black based edge detected image is better for the operator.

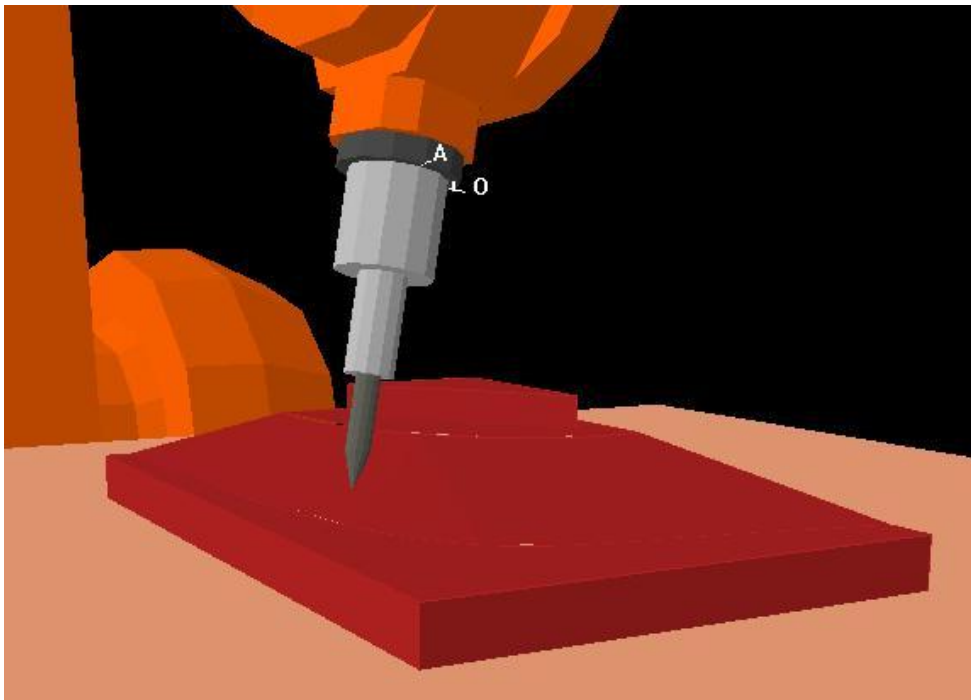


*Fig.5. Image processing sequence*

Previous work of authors (DIMAN: Distributed Image Analyzer) [6], made a stable background of the implementation of the image processing steps.

### 3.2 MAPPING FROM 2D TO 6D

The result of the previous section was a set of 2D coordinates. In this section these 2D coordinates will be transformed into 6D robot pose coordinates. This 2D-6D mapping is done within a standard offline programming tool where the CAD model of the work piece is introduced together with the robot. Fig.6. shows the offline environment, programmed in the IGRIP® simulation tool. IGRIP® is a powerful robot simulation tool, where complete robot manufacturing cell can be constructed and controlled in virtual reality.



*Fig.6. Offline programming environment*

Since the 2D track generator gives no information on the depth coordinate ( $z$ ) this is firstly established using an automatic collision (hit and fallback) procedure. A pen is attached to the robot arm, and the robot tries to reach the surface of the work piece from the predefined 2D coordinate. If the robot hits the surface, the  $z$  position is stored. These 3D positions are known as the cutting points  $p_w^n$  (index  $w$ = work piece reference coordinate system).

In each cutting point ( $p_w^n$ ) we determine the surface inclination in the feed direction  $x_n$  as

$$x_n = \frac{p_w^{n+1} - p_w^n}{|p_w^{n+1} - p_w^n|} \quad (3)$$

Surface inclination ( $y_n$ ) in a direction perpendicular to the feed direction is determined by

$$y_n = \frac{p_w^{n_2} - p_w^{n_1}}{|p_w^{n_2} - p_w^{n_1}|} \quad (4)$$

Then the surface normal ( $z_n$ ) in the cutting point is found as

$$z_n = x_n \times y_n \quad (5)$$

In each cutting point ( $p_w^n$ ) the directional cosines  $x_n$ ,  $y_n$ ,  $z_n$  forms a surface coordinate system  $K_n$ . To collect all parameters a ( $4 \times 4$ ) transformation matrix is created

$$T_w^n = \begin{bmatrix} x_n & 0 \\ y_n & 0 \\ z_n & 0 \\ p_w^n & 1 \end{bmatrix} \quad (6)$$

The matrix (6) represents the transformation between the cutting point coordinate ( $K_n$ ) system and the work piece reference coordinate system  $K_w$ .

Dependant on the shape and size of the cutting tool, as mounted in the robot hand and represented by the tool centre point coordinate system  $K_v$ , the most effective cutting conditions are achieved when the tool is aligned at certain angles to the surface of the work piece [7]. These angles are often referred to as the “lead” and “tilt” angles.

The lead angle ( $\beta$ ) is the angle between the surface normal ( $z_n$ ) and the tool axis ( $z_v$ ) in the feeding direction while the tilt angle ( $\alpha$ ) is the angle between the surface normal ( $z_n$ ) and the tool axis ( $z_v$ ) in a direction perpendicular to the feeding direction Fig.7. shows the lead and tilt angles.

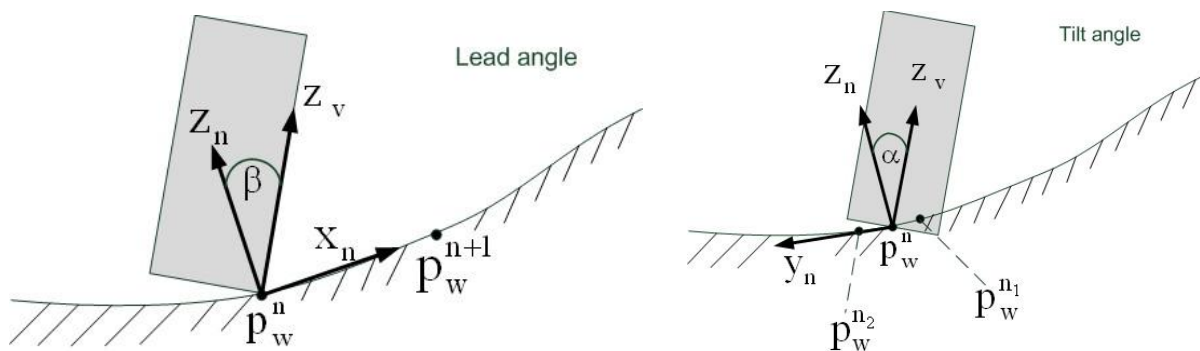


Fig.7. Lead and tilt angles

In each cutting point the existence of a lead and/or tilt angle modifies the cutting point coordinate system  $K_v$  and the belonging transformation matrix (6) according to the following

$$T_w^n = \begin{bmatrix} \cos \beta & 0 & -\sin \beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \beta & 0 & \cos \beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha & 0 \\ 0 & -\sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_w^n \quad (7)$$

A tool orientation angle  $\gamma$ , defined as a rotation around the  $z_v$  axis, can in some cases come handy when it is necessary to direct the cutting sparks away from the surface of the work piece or collect cutting chips in a suitable container mounted on the robot wrist. The  $\gamma$  angle adds to the transformation in (7)

$$T_w^n = \begin{bmatrix} \cos \gamma & \sin \gamma & 0 & 0 \\ -\sin \gamma & \cos \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_w^n \quad (8)$$

The transformation matrix (8) holds information on the position and orientation of the robot tool centre coordinate system  $K_v$  in each cutting point. According to (6) position data is extracted from the matrix as the last row in the matrix

The orientation is given by the directional cosines represented by the upper left ( $3 \times 3$ ) corner matrix of (8). This nine parameter matrix can be reduced to a minimum of three angular parameters dependant on the choice of the orientation convention. Most industrial robot systems operate with less than nine parameter description of orientation and a full conversion from nine to three parameters reduces the amount of necessary data input. In [2] details of most used orientation conventions and transformation are found.

After the above calculations the 6D coordinates are saved as robot (vendor dependant) coordinate file, still with reference to the work piece coordinate system.

### 3.3 WORK CELL TRANSFORAMTION MODULE

In the machining cell it is necessary to establish the relationship between the robot base and work piece coordinate system. The accuracy of the robot path will greatly depend on how well this work is carried out. The work cell transformation module undertakes these calculations.

As mentioned earlier the work piece coordinate system is set-up and defined from a selection of well known basic geometric elements (in the ideal work piece) and these elements is re-found on the actual work piece by using the robot as the measuring device. In general, the same methodology used for coordinate measuring machines should be used for the re-establishment of the work piece coordinate system. In [8] extensive recommendations are found.

After the work piece coordinate system is found the generated robot path is transformed into robot base coordinates and the program executed by the robot.

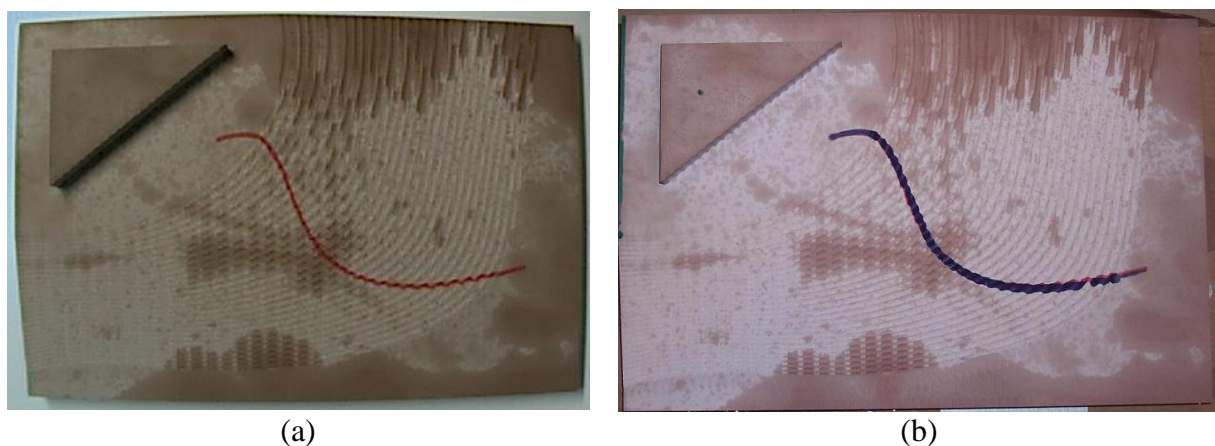
## 4. EXPERIMENTAL RESULTS

Some initial experimental tests have been carried out on the proposed system.

First, to measure the effect of the barrel/pincushion distortion and chromatic aberration of the cameras, photos were taken of three elementary geometrical objects (drawn onto a "millimeter" paper). Two different kinds of cameras were used for this test. The first camera

was a web camera (640\*480 resolution) and the second camera was a typical compact digital camera (2856\*2142 resolution). Pictures were taken from the same height and enlarged to the same resolution. In the 2D track detector, after the operator had identified the three geometrical objects, the generated 2D path was matched with the original geometrical objects. The mean deviations between the original and captured images were approximately 1 mm for the web camera and 0.5 mm for the compact digital camera. As expected, the accuracy is better for the higher resolution camera.

Secondly, to measure the complete error chain, a set of experiments have been carried out with an ABB IRB 2000 robot. For these tests the low resolution web camera were used to capture a hand drawn line on a sculptured surface (work piece). After the image processing the robot was instructed to follow the generated path. Fig.8. shows the ideal hand-drawn path (a.) and the resulting path (b.) executed by the robot.



**Fig.8.** System test

The deviation between the hand drawn and the robot generated answer was approximate 1 mm, the same accuracy as for the web camera. A machining accuracy of less than a millimeter is acceptable in grinding and deburring operations.

## 5. CONCLUSION

In this paper, a further development, of our vision based robot programming methodology has been introduced. The proposed system makes robot programming easy, rapid and flexible. The man-machine interaction is very human friendly and the system captures the expertise of an experienced operator in grinding/machining.

The proposed system is accurate enough for most grinding and deburring operations.

## ACKNOWLEDGEMENTS

This work was supported in part by the Narvik University College and the Budapest University of Technology and Economics [3].

## REFERENCES

*Books:*

- [1] S. Kalpakjian , S. R. Schmid,(2006) "Manufacturing Engineering and Technology," Fifth Edition. Pearson Education, Inc. Pearson Prentice Hall. Upper Saddle River, NJ 07458 ISBN 0-13-148965-8

- [2] J.J. Craig, (2005), "Introduction to robotics, Mechanics and control" third ed. Pearson Education, Inc. Pearson Prentice Hall. Upper Saddle River, NJ 07458, ISBN 0-13-123629-6
- [3] G. Sziebig, (2007) "Interactive vision-based robot path planning" Final project, Budapest University of Technology and Economics and Narvik University College, Hungary/Norway

*Conference proceedings:*

- [4] B. Solvang, P. Korondi, G. Sziebig and N. Ando.,(2007) "SAPIR: Supervised and Adaptive Programming of Industrial Robots," in Proc. 11th International Conference on Intelligent Engineering Systems (INES'07), pp. 281-286 ISBN: 1-4244-1147-5.
- [5] T. Thomessen, T. K. Lien, and B. Solvang, (1999), "Robot control system for heavy grinding applications," in Proc. 30th International Symposium on Robotics, pp. 33-38
- [6] G. Sziebig, A. Gaudia, P. Korondi, and N. Ando, (2006), "Video image processing system for rt-middleware," in Proc. 7th International Symposium of Hungarian Researchers on Computational Intelligence (HUCI'06), pp. 461-472.
- [7] S. Köwerich, (2002) "5 akse maskinering, En state-of-the-art- studie", SINTEF report STF 38 A97247, Norway, ISBN 82-14-00695-3
- [8] K. Martinsen, (1992), "Kompendium i måleteknikk og vektorielle toleranser" Institutt for vekstedteknikk, Norges Tekniske Høgskole, Norway.