A Rapid Deployment Automation Solution for Robot Vision Applications

Florin Daniel ANTON, Theodor BORANGIU, Silvia ANTON

Abstract: In conventional robot system development different robot parts (sensors, processing elements and actuators) are combined together in a compact, self contained system. The need for faster development and deployment, system reconfigurability and flexibility required the introduction of rapid deployment automation for robot systems in flexible manufacturing cells. Vision sensors are one of the most important sensors in robot systems. When constructing a new robot system, it is desirable that vision and image processing components are just as easily integrated as any other robot components. In some situations robot stations must be upgraded with vision systems in order to accomplish new tasks or to improve current work. In this case the Rapid Deployment Automation concept is the base to solve such problems in a shorter time with the smallest impact on the production flow in terms of production downtime. The paper describes an RDA solution of vision upgrade for robot stations in a production plant which produces objects made of porcelain.

Key words: Rapid Deployment Automation, Robot-Vision, open contour object model, flexible manufacturing cells, shared robot workspace,

1 Introduction

The paper describes a process of upgrading a plate production line which uses ABB robots by adding artificial vision systems.

The production line uses raw materials which feed a stencil. The stencil applies pressure on the material and result the moulds (raw material has the shape of the plates). Before the plates are transported to the oven for the baking process, they are routed to multi-robot stations where robots are used for polishing the plates and remove excess material (small irregularities on the edges of the plate).

Dr. Florin Daniel Anton, Prof. Theodor Borangiu, Dr. Silvia Anton, Department of Automation and Industrial Informatics, University Politehnica of Bucharest, Romania

The stations use groups of 4 ABB robots. The problem is that the plates are not stopped precisely in the same position each time the robots take them to be polished. Due to this problem a percentage of plates are polished incorrectly which results in damage of the plate edges.

In these conditions a solution based on artificial vision has been considered (see *Figure 1*).

Each station must have a vision system which locates the plate with precision and sends the coordinates to the robots. The task is more difficult than implementing the robot-vision (RV) solution from the start because now production is interrupted, and

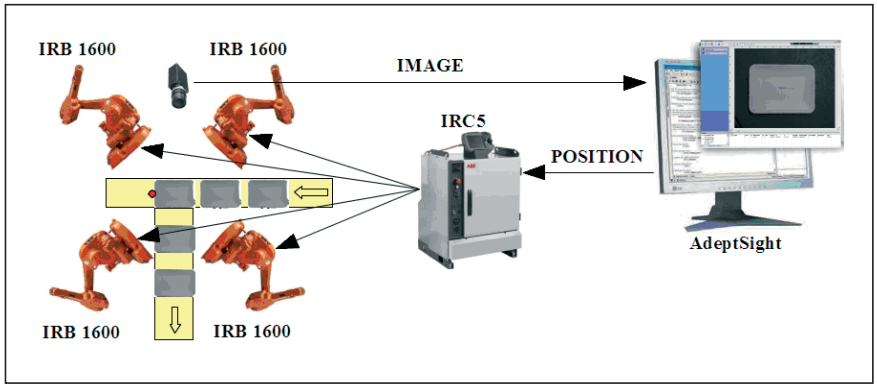

Figure 1. *Vision integration in the robotized production line*

this time must be as short as possible.

Furthermore, the vision task is very difficult due to environmental conditions; firstly the inspected object has a shape which, due to the fact that it is based on interior shadows of small canals included in the plate, cannot be viewed entirely, secondly the environment is heavy loaded with dust which comes from the plate polishing operations.

2 The Vision System

For the vision solution, few systems have been considered, and finally AdeptSight was chosen for multiple reasons:

- The vision system have high performance in object localisation: 1/40 of a pixel in position, and 0.01 degree in rotation
- The system has the possibility to train and edit object models based on non connected contours
- The programming interface is based on visual tools (visual programming) for rapid development, and also has the possibility to be integrated with high level programming languages (C#) for complex applications.
- The system can be easily integrated with any robotic system (see *Figure 2*).

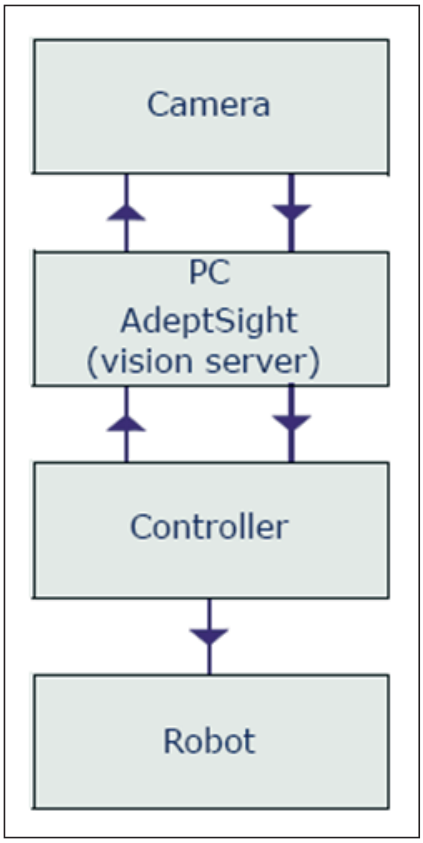

Figure 2. *Robot-Vision Integration*

In Figure 2 the integration diagram for the AdeptSight vision system in robotic applications is presented. AdeptSight uses up to four FireWire cameras connected directly to the PC where the AdeptSight software is installed.

The development of vision applications is based on vision projects which have the following structure (see *Figure 3*): The vision project is separated into two main parts, the first part handles the hardware and the communication environment and is composed of communication routines and a configuration of system devices like cameras (Basler, Direct Show or Emulation – virtual camera), robot controllers (native communica-

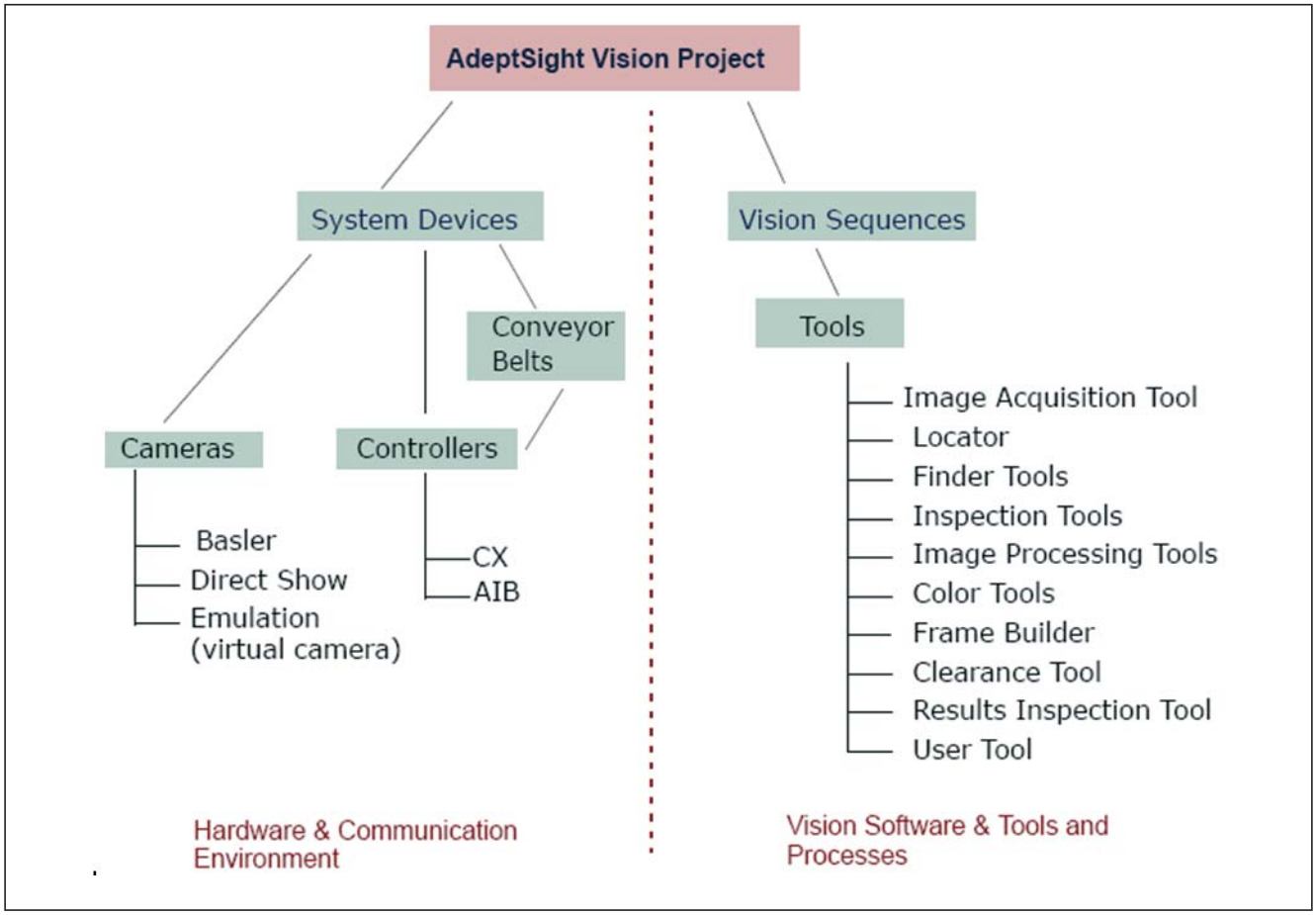

Figure 3. *The vision project structure*

tion support with Adept CX and AIB controllers) and conveyor belts.

The second part is represented by the vision sequences, which compose the principal part of the project. The sequences are composed of vision tools connected together and executed step by step in a sequence defined by the programmer. In addition, the user can develop C# programs which interact with the AdeptSight project and extend their functions regarding the communication with other robot systems and other functions which are not implemented in AdeptSight.

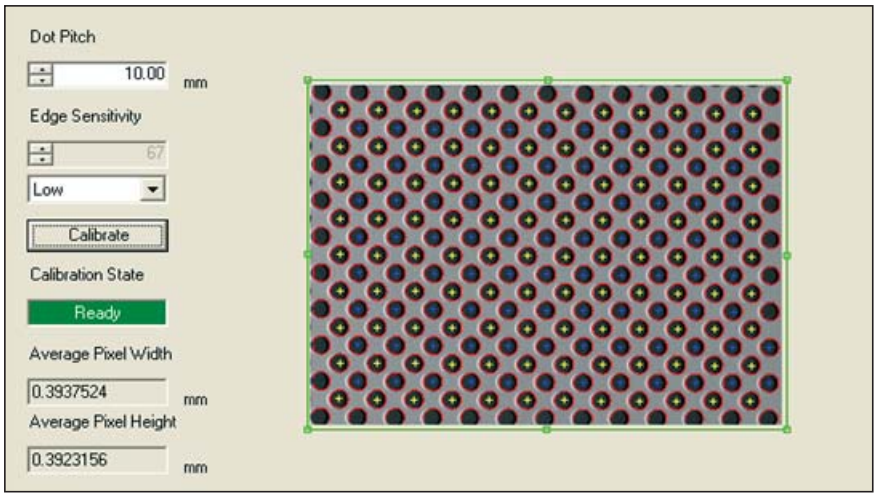

Figure 4. *The result obtained after calibration*

3 Vision Project Implementation

The vision is calibrated using a dotted pattern which is placed in front of the camera on the working surface. The calibration is executed using a 2D camera calibration wizard which guides the user step by step through the calibration process. The single information which the user must supply is the Dot Pitch of the calibration pattern, the rest of the process

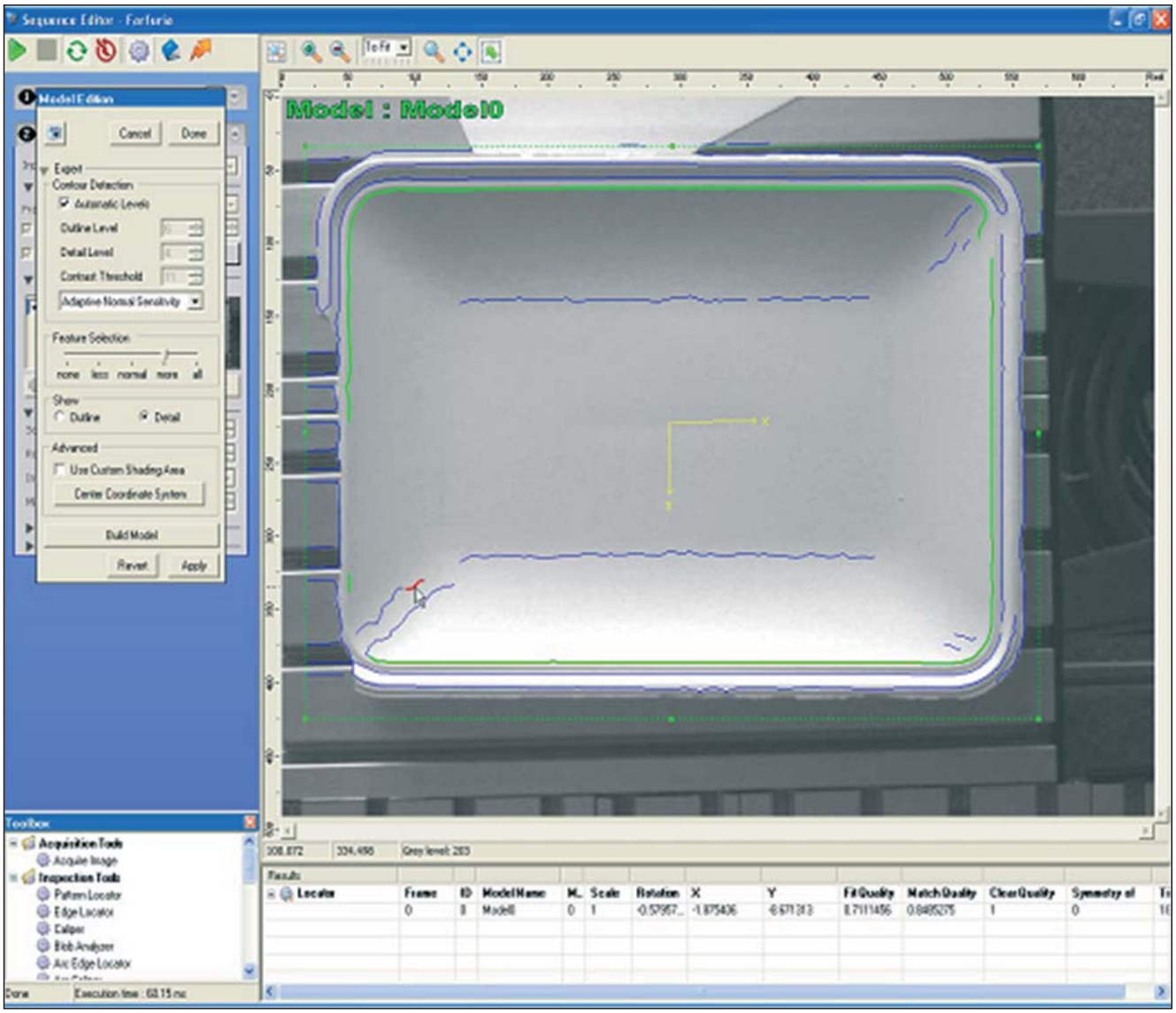

Figure 5. *The plate model, using internal, open contours*

is handled by the wizard. After the 2D camera calibration the following information's are obtained:

- Average Pixel Width/Height (*Figure 4*)
- The lens distortions are corrected
- The perspective distortion is also corrected

After the 2D camera calibration, the sequence can be loaded into the project, and for this application which requires only object recognition, the sequence is very simple and consists of only two vision tools; an acquisition tool which obtains the image from the Basler (640x480) camera, and a second tool named Locator which has recognizes the object.

In this application the difficulty is to detect the correct position of the plate disregarding the outer irregularities which are generated by the fabrication process.

Due to the fact that the contour features are contained inside the object it is in this case very difficult to create a lighting setup in such a way that the image of the object contains the complete internal contours. AdeptSight allows the user to generate the object model using open contours, and also gives the possibility to edit (remove/add) parts of the contours which are of interest.

In *Figure 5* the interface for model editing is presented. The system, based on the contrast threshold, and the outline and detail levels detects the contours and proposes the user a set of contours for model building. The outline level provides a coarser level of contours than the detail level. The location process uses outline level contours to rapidly identify and roughly locate potential instances of the object and then, the location process uses the detail level contours to confirm the identification of an object instance and refine its location within image.

The user can modify the selected contours by deleting/adding contours, or select only parts of the proposed contours [1], [2]. In Figure 5 the red line is selected for deletion, the blue contours are marked as deleted (will not be used for model building) and the green contours are valid contours. After all contours have been selected, the model can be build and used for recognition.

4 Robot-Vision Integration

In order to integrate the vision system with the robot two tasks must be achieved [4]:

1. A robot-vision calibration procedure must be created and executed.
2. A robot-vision server communication mechanism must be implemented.

4.1 Robot-Vision calibration

The robot-vision calibration process and the training of the object grasping model have been integrated in a single procedure. The procedure consists of four steps where the robot grasps the object and places it in different positions in the workspace for image acquisition and processing [5]:

1. The calibration object is placed in the workspace and grasped by the robot and then released (position P1). After the robot clears the vision plane the position of the object in the vision plane is computed by AdeptSight.
   The position of the object in point P1 is also computed in the vision plane and has the coordinates $P1'_{Xv}, P1'_{Yv}$ (the position of the coordinate system attached to the object model).
2. In the second step the robot grasps the object and places it in the same position, but rotated with 180° point P1′, and in the vision coordinates $P1'_{Xv}, P1'_{Yv}$. By comparing the position of the coordinate system of the object in these positions, the system can compute the position of the centre of the object, which in this case is the centre of mass of the model relative to the grasping point. In this case the grasping point is located in the image in the middle of the segment $[P1, P1']$ (see *Figure 6*).

In this case the grasping point coordinates in the vision space are given by:

$$P_{gvis}\begin{cases} P_{gvisx} = \min(P1_x, P1'_x) + \left|P1_x - P1'_x\right| \\ P_{gvisy} = \min(P1_y, P1'_y) + \left|P1_y - P1'_y\right| \end{cases} \quad (1)$$

where P_{gvis} is the grasping point in the vision workspace.

3. Next the object is placed in the position P2 which is trained relative to the position P1 shifted with 100 mm on X axis of the base coordinate system of the robot.
4. In the final step the robot places the object in the position P3

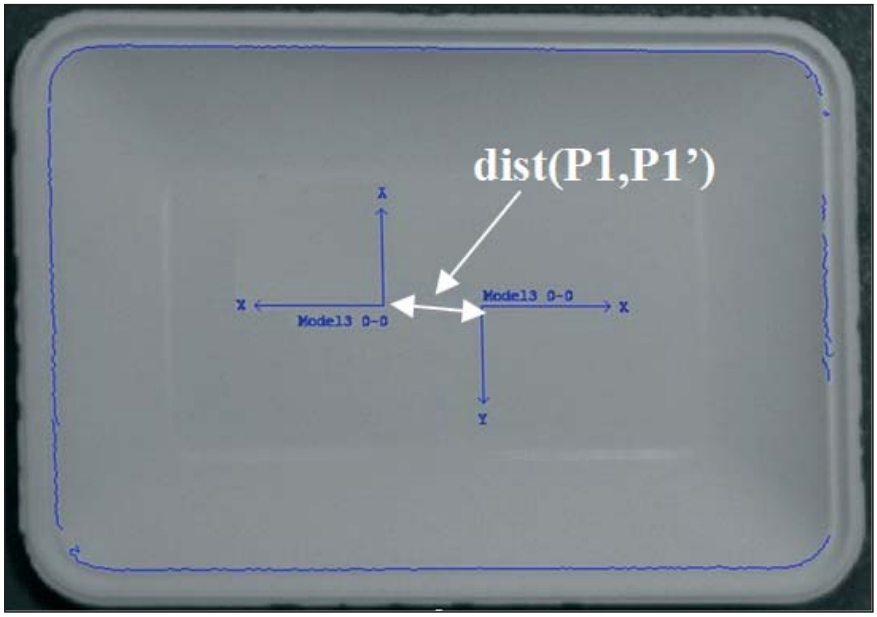

Figure 6. *The relationship Robot point – vision point*

which is trained relative to the position P1 shifted with 100 mm on Y axis of the base coordinate system of the robot. By knowing the correspondence robot-point – image-point, the system can now compute the orientation of the vision plane relative to the robot base coordinate system, and can also compute the distance which the robot must cover to reach the object which is placed at a certain distance from the initial point P1 in the image plane.

This can be expressed as:
For 100mm travelling length on the X coordinate system (base coordinate system) the object moves in the image $P2_x - P1_x$ on X_{vis} axis, and $P2_y - P1_y$ on Y_{vis}, the same travelling length on Y coordinate system generates $P3_x - P1_x$ on X_{vis} axis, and $P3_y - P1_y$ on Y_{vis}, on the vision workspace. Thus, the vision system is rotated with the angle:

$$\alpha = a\tan 2(P2_Y - P1_Y, P2_X - P1_X) \quad (2)$$

towards the base coordinate system. Therefore, for an object which is recognized in the image at the location P_v, in the world coordinate system the object will be grasped at the coordinates:

$$\begin{aligned} P_x &= P_{G\_x} + \sqrt{(P1_{\_x} - P_{v\_x})^2 + (P1_{\_x} - P_{v\_x})^2} \\ &\cdot \cos(\alpha + a\tan 2(P_{v\_y} - P1_{\_y}, P_{v\_x} - P1_{\_x}) \\ P_y &= P_{G\_y} + \sqrt{(P1_{\_x} - P_{v\_x})^2 + (P1_{\_x} - P_{v\_x})^2} \\ &\cdot \sin(\alpha + a\tan 2(P_{v\_y} - P1_{\_y}, P_{v\_x} - P1_{\_x}) \\ P_{rot} &= P_{G\_rot} + (P_{v\_rot} - P1_{\_rot}) \end{aligned} \quad (3)$$

where P_x, P_y, P_{rot} are the coordinates and the rotation of the grasping point of the object which was located in the vision workspace at the location P_v $(P_{v\_x}, P_{v\_y}, P_{v\_rot})$, P_G $(P_{G\_x}, P_{G\_y}, P_{G\_rot})$ is the grasping point (in the centre of the mass of the object in the base coordinates system) for the object located in the image in P1 $(P1_{\_x}, P1_{\_y}, P1_{\_rot})$.

4.2 Robot – Vision server communication

The application runs two communication threads; a TCP/IP server and also has a serial communication thread. Both threads have the same role to listen. If they receive an acquisition request they initialize the execution of the AdeptSight sequence, returning three numbers specifying the position and orientation of the plate (X, Y in mm, and the angle in degrees). The position and orientation is specified relative to the calibration object.

The requests are sent as ASCII characters, and they are of two types (i – information for debugging, or r – real requests), when the vision server receives a request, the vision sequence is executed, the object is recognized based on the internal contours, and the values (X, Y and rotation) relative to the initial grasping point (from the calibration procedure) are computed and sent to the robot. In *Figure 7* the output in a telnet window can be seen.

When the robot receives the three values, it shifts the initial grasping position (from the calibration) and grasps the plate.

The following code must be part of the main program in order to integrate the communication with the vision server [1]:

```
!Open the communication channel (Serial line)
Open "COM1:", COM1 \Bin;
ClearIOBuff COM1;
WaitTime\InPos, 0.5;

!Make a request
WriteStrBin COM1,"r";

!Read the data streams
XData:= ReadStr (COM1 \Delim:="/");
YData:= ReadStr (COM1 \Delim:="/");
AngleData:= ReadStr (COM1 \Delim:="/");
x1:=nXOffs;
y1:=nYOffs;
r1:=nAngle;

!Parsing the data streams
bOK:=StrToVal(XData,nXOffs);
bOK:=StrToVal(YData,nYOffs);
bOK:=StrToVal(AngleData,nAngle);

IF (x1 = nXOffs) AND (y1 = nYOffs) AND (r1 = nAngle) THEN

!Computing the grasping position
pos:=RelTool(Offs(p1,nXoffs,nYOffs,0), 0, 0, 0 \Rz:= -nAngle);
ENDIF
……..
!Closing the communication
Close COM1;
```

5 Conclusions

The goal of the RDA is to reduce the total time, risk, and cost required conceptualizing, designing, engineering, and implementing intelligent automation systems [6].

The topic addressed in this paper was the RDA concept, implemented in a robotized manufacturing plant.

The main goal was to introduce the vision in an already installed solution (without vision), the solution which must be stopped a very small amount of time for the vision upgrade.

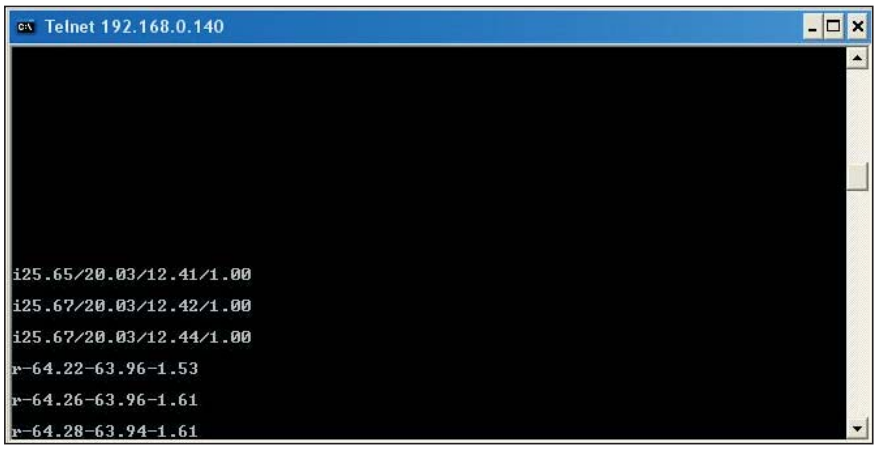

Figure 7. *Communication streams between RV*

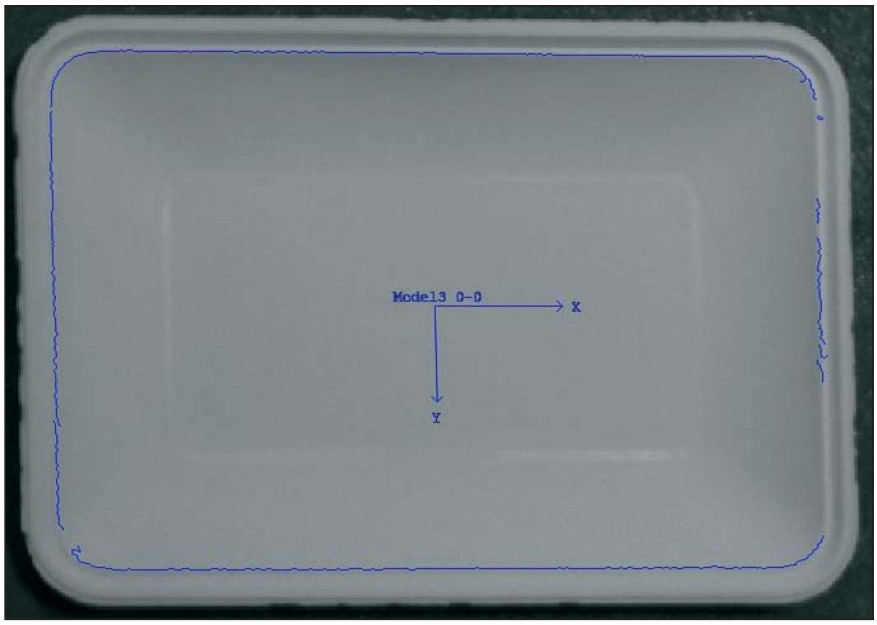

Figure 8. *A recognized plate*

Another problem was choosing the vision system, which had to be a vision system capable of recognizing objects (see *Figure 8*) whose models must be editable and based on interior contours.

The selected vision system was AdeptSight which is robust enough and has the requested performances, and can also be easily integrated with other industrial equipments.

The vision system allows a rapid development of the application based on visual tools which can be combined and configured resulting sequences which can be executed from external C# applications.

Our application was based on C# and managed the robot – vision communication and sequence execution. Also the robot-vision calibration procedure and the grasping model were solved by this application.

The approached solution demonstrates the RDA concept and can be considered as a p.o.f. (proof of concept) in this field.

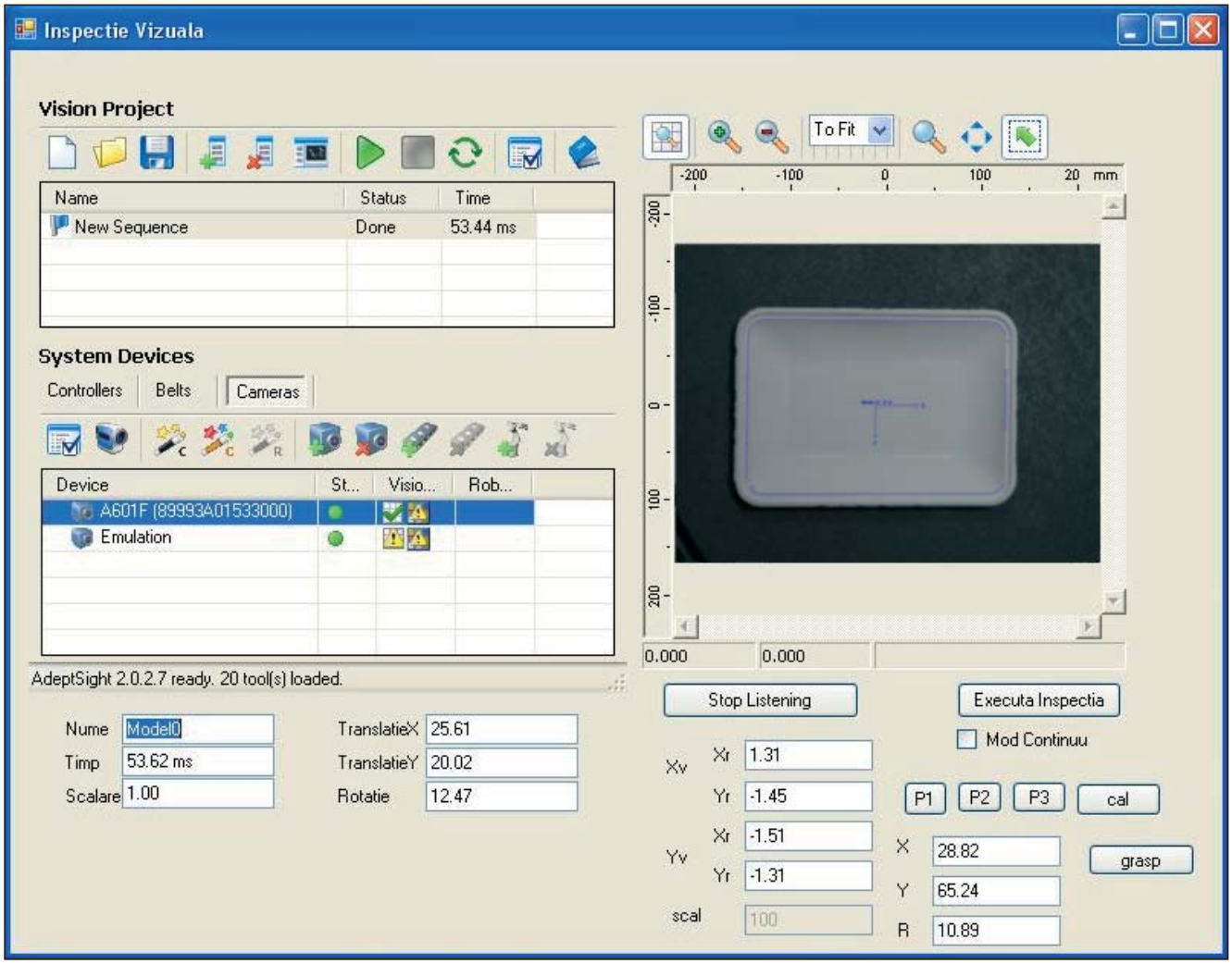

Figure 9. *The application interface*

Using advanced programming tools and the RDA equipment allowed to design, build and integrate an elegant, user friendly solution (see *Figure 9*) and helped to remove the problems in the production plant and to raise productivity and profit in a very short time. The rate of rejected plates due to bad object grasping and polishing was reduced drastically to almost 0%, the number of discarded plates at the polishing stations was not 100% reduced due to other factors such as belt vibrations and due to a relatively weak material of the plates (when they are removed from the stencil).

Sources

[1] ABB, 2004: Technical Reference Manual, RAPID Instructions, Functions, and Data types.

[2] Adept Technology Inc., 2001: Adept-Vision Reference Guide Version 14.0 Part Number 00964-03000, San Jose, Technical Publications.

[3] Adept Technology Inc, 2007: Adept-Sight 2.0 Online Help, San Jose, California, USA.

[4] Anton, S.: Integrating multiple robot-vision systems in intelligent assembly/disassembly structures. PhD thesis, UPB Automatics and Industrial Informatics Department, 2008.

[5] Borangiu, Th.: Intelligent Image Processing in Robotics and Manufacturing, Bucharest, Romanian Academy Press, 2004.

[6] Dunn, A.: Adept's unveils rapid deployment automation partners. Manufacturing Automation. Vital information publications, 1996.

The article was originally published on xxth RAAD in Bucharest, Romania

Hiter razvoj avtomatiziranih sistemov z uporabo računalniškega vida

Razširjeni povzetek

V običajno uporabljenih načinih razvoja robotiziranih sistemov se sestavi večje število robotskih komponent, kot so senzorji, komponente za krmiljenje in izvajanje procesov in aktuatorji, v kompaktni dokončni sistem. Potrebe po hitrem razvoju in postavitvi, možnosti preurejanja in ponovne uporabe kakor tudi večje zahteve po fleksibilnosti zahtevajo hitro se razvijajočo avtomatizacijo za robotske sisteme v fleksibilnih izdelovalnih celicah. Pri tem je robotski vid ena izmed ključnih komponent robotskega sistema, zato je zaželeno, da je mogoče robotski vid kakor tudi procesiranje – obdelavo slike – integrirati v robotski sistem čim enostavneje. To velja tako za razvoj novih kot za nadgraditev obstoječih robotiziranih celic. Pri tem je koncept hitrega in avtomatiziranega razvoja – rapid deployment automation (RDA) – osnova za hitro doseganje rezultatov in čim manjši vpliv na zaustavitev in ponovno uporabo robotskega sistema v proizvodnji. V prispevku bo prikazana uporaba koncepta RDA pri nadgraditvi robota ABB za izdelavo plošč z računalniškim vidom. Pred dodajanjem plošč na tiskanje se te polirajo in odpravijo manjše napake na površini. Postajo sestavljajo štirje roboti ABB, ki primejo ploščo in jo držijo med poliranjem. Zaradi nenatančnega pozicioniranja jih robot ne prime pravilno, kar je vzrok za nekakovostno poliranje. Da bi se temu izognili, je bilo treba namestiti k robotski celici umetni vid, kar omogoča zmanjšanje vpliva nenatančnosti v pozicioniranju na kakovost poliranja (*slika 1*).

Za ta namen je bil kot ustrezna rešitev izbran AdeptSight, saj ima dovolj dobre lastnosti pri lociranju objektov. Programsko orodje je osnovano na optičnih orodjih za hiter razvoj in možnost povezave s programskim jezikom za ta kompleksni primer uporabe. Sistem je mogoče enostavno povezati z robotskim krmiljem (*slika 2*). Pri razvoju in vgraditvi robotskega vida je bil uporabljen sistematičen pristop RDA, ki je razdeljen v dva dela. Prvi vključuje razvoj strojne opreme in komunikacijskega okolja in ga sestavljajo različne komunikacijske rutine, kamere, krmilniki robota in transportnega traku. Drugi del predstavlja zaporedje aktivnosti in različna orodja (*slika 3*). Razvoj in postavitev vključujeta več aktivnosti, med njimi tudi kalibracijo robotskega vida in razvoj komunikacije med vidom in robotom.

Cilj RDA je bil zmanjšanje skupnega časa obdelave plošč in zmanjšanje napak ter hkrati zmanjšanje stroškov za izdelavo koncepta, načrta in integracijo inteligentnega avtomatiziranega sistema. Uporaba sistematičnega pristopa RDA pri razvoju primera uporabe omogoča hitro in prijazno delo za inženirje. Projekt dograditve umetnega vida k robotski celici je bil izveden hitro in po končani prenovi ni bilo več napak zaradi napačnega pozicioniranja.

Ključne besede: hiter razvoj avtomatizacije, robotski vid, odprti model oblike objektov, flesibilne izdelovalne celice, skupni delovni prostor robota,