Robotized Painting with Automatic Reconfiguration

Marcos Ferreira, Paulo Malheiros , A. Paulo Moreira and Norberto Pires

***Abstract*— Industrial manipulators are widely used on production lines due to its highly accurate movements, In this paper, a case study of an adaptive robotized painting system for small production series is presented. The concept is based on contactless technology, using artificial vision and laser scanning, to identify and characterize different objects traveling on a conveyor. The collected data enables automatic system reconfigurations according to the specific profile of each object. A robotic manipulator executes the painting process after its base algorithms have been adapted online, and the system becomes fully autonomous and capable of dealing with small production series without human intervention for reprogramming and adjustments. Described methodology can be applied to numerous applications, other than painting and object recognition.**

I. INTRODUCTION

A. Motivation

Production lines tend to evolve into the concept of mass customization, i.e., working on small series with adapted and specialized procedures to each of them according to costumer specific needs. High versatility is mandatory in these systems and robotized cells demand additional efforts to be integrated in such systems: industrial manipulators still take a long time to reconfigure. Programming is truly time consuming and usually require experienced and highly qualified workers. Overall not compatible with flexible setups neither with companies budgets since both qualified programmers and reconfigurations associated downtime imply strong financial efforts.

Despite these, manipulators are strongly desired at production lines due to a series of advantages over human work, e.g., the ability to work continuously, high accuracy and repeatability, immunity to fatigue, distractions and even hazardous environments.

Taking the case study of an industrial painting system working on small production series, a flexible architecture is presented that enables fast system reconfigurations and adaptive behavior without human intervention.

B. Proposed Solution: Architecture and Technologies

The developed system integrates three different fields (Fig. 1).

First, we make use of an artificial vision system that captures images of the different pieces traveling along a conveyor. Together with a line-laser that scans the entire pieces as they are transported (at constant speed), 3D models of the pieces are built.

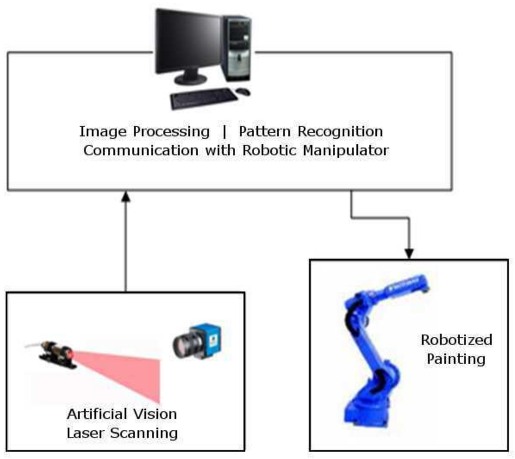
Image Processing | Pattern Recognition
Communication with Robotic Manipulator
Artificial Vision
Laser Scanning
Robotized
Painting

Fig. 1. System architecture

Second, from the 3D models, several algorithms are run to extract information on the objects — size, boundary, texture, orientation ... — and a simple machine learning algorithm is used to classify the piece (the different kinds of parts are known *a priori*).

At last, all the data is transmitted to the industrial manipulator in charge of painting and it adapts its painting schemes to match the piece's size and layout.

C. Related work

Even though the integration of artificial vision with laser triangulation, pattern recognition and flexible reprogramming schemes of industrial manipulators isn't found yet in the literature, at least for all these fields together, the proposed method of finding 3D models has been largely discussed. A lot of research has been carried out with facial recognition [7], object dimensions measurements [8] and even in the field of inspection for quality control [9]. Some analysis on precision have already been made, comparing the use of single or multi laser beams [10] or alternative computer vision systems, as stereoscopic pairs [11]. Off-the-shelf technologies can also be option but, generally, these solutions are highly expensive when compared to the custom setup presented in this paper.

II. COMPUTER VISION AND LASER TRIANGULATION

The artificial vision subsystem is responsible for capturing images of the objects on the conveyor, on which a line-laser

M. Ferreira, P. Malheiros and A.P. Moreira are with Faculty of Engineering, Department of Electrical and Computer Engineering, University of Porto, and with INESC-Porto, Portugal `[maf,plm,amoreira]@fe.up.pt`

N. Pires is with the Mechanical Engineering Department, University of Coimbra, Portugal `jnp@robotics.dem.uc.pt`

is projected. This line is identified in each video frame and it generates three dimensional information about each piece.

First line of image processing starts with isolating the laser line in each frame: the environment illumination is controlled (this makes the area the camera is filming dark) and this way the laser line appears brighter in the images. A simple binarization algorithm is applied

$$F_{B\&W}[u,v] = \begin{cases} 0 & \text{if } F[u,v] < \text{threshold} \\ 1 & \text{if } F[u,v] \geq \text{threshold} \end{cases} \tag{1}$$

and we now work over very clean images as the following one:

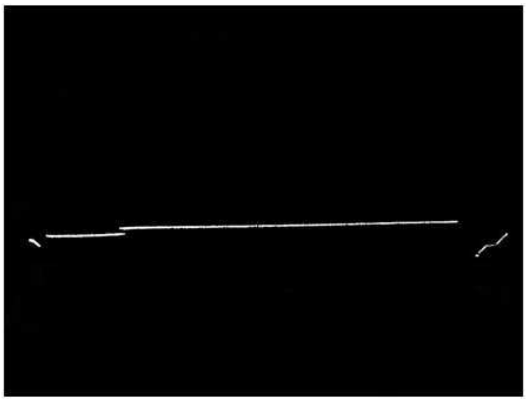

Fig. 2. Result of the binarization process

A. Camera and Laser Calibration

Calibration is the method of identification of our artificial vision system parameters. It allows us to recognize the position in the world of any pixel of an image or vice-versa.

1) Distortion Compensation: In order to achieve better results in the calibration process, we start off by compensating the barrel effect distortion of the camera. This phenomena can be modeled as a radial effect [3] and approximated by

$$\begin{bmatrix} u_R \\ v_R \end{bmatrix} = \left(1+kr^2\right) \begin{bmatrix} u_D \\ v_D \end{bmatrix} \tag{2}$$

meaning the further we step away from the center of the image the more it is distorted, images get spherised at their center. In (2), k represents the barrel distortion coefficient, r the distance to the center of the image, and subscripts D and R refer to original and corrected coordinates respectively. Estimating k is trivially accomplished using the image of a straight line: it should appear round in the image due to barrel effect so, after sampling some pixels of the line, assuming the true values should belong to a straight line, one can use (2) to do a simple regression on k.

Next we fit the system into a model.

2) Camera Model: The most widely used model for conventional cameras is the *pinhole* model [1]. The phenomena of light passing through a tiny hole and being projected into a plane, which is the fundamental of *pinhole* model, can be synthetically described as shown in Fig. 3;

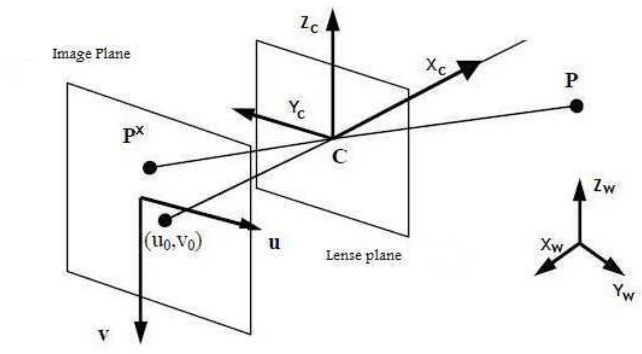

Fig. 3. Pinhole model and coordinate systems specification [2]

Take all coordinate systems as Cartesian. In homogeneous coordinates, the relation between the pixels coordinates and the world coordinates is given in [2]:

$$P_x = H \cdot P \tag{3}$$

$$H = H_I^F H_F^C H_C^W \tag{4}$$

where H is the projection matrix of the camera and results of the composition of 3 homogeneous transformations – H_C^W mapping world coordinates in the frame defined by the camera, H_F^C representing a perspective projection from the 3D camera frame into the 2D image frame, and finally H_I^F which scales the last transformation according to the image size. These are, respectively, defined as:

$$H_C^W = \begin{bmatrix} R_C^W & T_C^W \\ 0 & 1 \end{bmatrix} \tag{5}$$

R_C^W is a rotation matrix and T_C^W the translation vector;

$$H_F^C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & f^{-1} & 0 \end{bmatrix} \tag{6}$$

f is the focal length, the distance from the image plane and the lens plane;

$$H_I^F = \begin{bmatrix} du & 0 & u_0 \\ 0 & -dv & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where (u_0,v_0) is the origin of the image frame and du,dv are the size of one pixel.

An important note on (3): the system is not invertible meaning that one point in the world has one and only one pixel in the image where it can appear but an arbitrary pixel may be the projection of an infinity amount of points of the world, yet all of them collinear.

3) Camera Calibration: Point-to-Point Correspondence : From (5),(6) and (7) we can see that there are 11 parameters we need to estimate to fully calibrate the system.

To simplify the problem, we can do the following assumptions:

- Assume pixels are squares and get two parameters from the camera's datasheet [5]: the width and height of the pixels, du, dv
- Assume the center of the image will be in the central pixel: v_0, u_0 are half the vertical and horizontal resolution of the camera, respectively
- The camera position in known (T_C^W), actually it can be measured manually

Since camera position takes 3 parameters, translation in x,y and z , we are left with only 4 parameters to estimate: camera rotation in all 3 axis and focal length.

To estimate these, a point-to-point correspondence algorithm was implemented using only 2 points. We start off by choosing one of the calibration points, P_{c1}, as the one that appears in the central pixel of the image — $P_{c1}^x = (u_0, v_0)$ — and we manually measure its position in the world (x_{c1}, y_{c1}, z_{c1}) .

The $OX_cY_cZ_c$ reference frame from Fig. 3 is obtained from $OX_wY_wZ_w$ with 3 rotations $(R_{\theta x}, R_{\theta y}, R_{\theta z})$, 2 of which we can already define using the point P_{c1}. From the scheme presented in Fig. 4, it's trivial to write the relations:

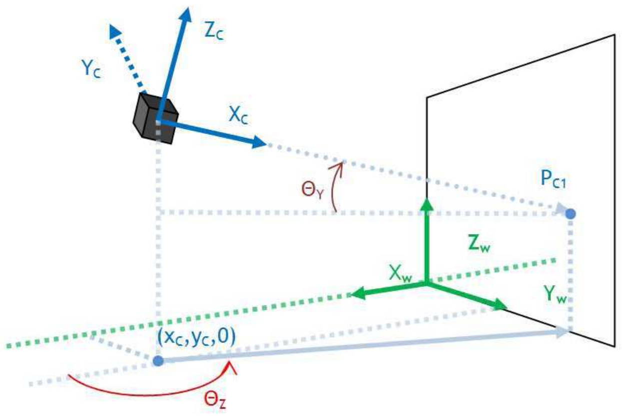

Fig. 4. Relation between world and camera referencials

$$\begin{cases} \theta_z = \arctan\left(\frac{y_{c1}-y_c}{x_{c1}-x_c}\right) \\ \theta_y = \arctan\left(\frac{z_c - z_{c1}}{\sqrt{(x_{c1}-x_c)^2+(x_{c1}-x_c)^2}}\right) \end{cases} \tag{8}$$

Estimating the last rotation angle, $R_{\theta x}$, requires the second calibration point, P_{c2}. For this one, choose it away from P_{c1} since this way robustness is improved, measure its position and then see its coordinates in the image (or vice-versa).

We now make use of (3): we compute P_{c2}^x (the position on the image of P_{c2}) taking the last estimates we have on $R_{\theta y}$ and $R_{\theta z}$ (obtained from (8)) and making $R_{\theta x} = 0$ — focal length is given a random value and it will become clear why later on. Comparing this estimation with the real value (remember that P_{c2} is known, both in world and image coordinates), the real value of θ_x can be computed: it's simply the angle, in the image plane, between the real position and our estimate with $\theta_x = 0$.

The last parameter left to estimate is the focal length. In the image plane, different focal lengths mean different distances to its center and it's why it has no influence over the estimation of the rotations. Then again, using (3) and point P_{c2}, we compute $\left\|u_{c2}^{(r)}, v_{c2}^{(r)}\right\|$ (the distance to the origin of the image, according to our model, with random focal length f_r). Focal length is determined by comparing this value with the real one we know $\|u_{c2}, v_{c2}\|$. The ratio of these two quantities can be applied to our random value of focal length so that the true value is found:

$$f = f_r \frac{\left\|u_{c2}^{(r)}, v_{c2}^{(r)}\right\|}{\|u_{c2}, v_{c2}\|} \tag{9}$$

4) Laser Calibration: In order to use the inverse of (3), the camera information is complemented through the use of the line-laser. See Fig. 5.

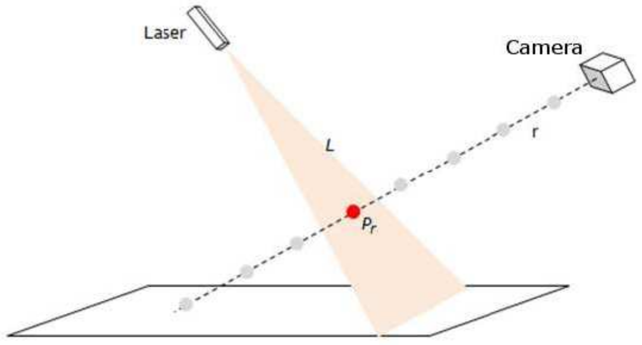

Fig. 5. Intersection of laser plane with the half-line containing all the points that are projected into the same pixel

One might consider that the line-laser actually originates a plane, L. The intersection of this plane with the half-line obtained by $P = H^{-1}P^x$ results in a single well defined point in space — note that the laser must be placed obliquely in relation to the camera.

From the equations of both the half-line, r, and the plane, L :

$$r: \ P_r = P_{r0} + w_r t \ , \ t \in \mathbb{R} \tag{10}$$

$$L: \quad w_n(P_L - P_{L0}) = 0 \tag{11}$$

where P_{r0} and P_{L0} are known points from the line and the plane respectively (e.g, the position of the camera, where the half-line starts, and the position of the laser that also belongs to the plane); $w_r = \begin{bmatrix} x_r & y_r & z_r \end{bmatrix}^T$ is the vector with the direction of the half-line and $w_n = \begin{bmatrix} x_n & y_n & z_n \end{bmatrix}^T$ is a vector orthogonal to the laser plane.

Finding the point that belongs, simultaneously, to r and L leave us with

$$(P_{r0} + w_r t - P_{L0})w_n = 0, \tag{12}$$

and then we get t as

$$t = \frac{d - x_n x_{r0} - y_n y_{r0} - z_n z_{r0}}{x_n x_r + y_n y_r + z_n z_r}\ ,\ d = w_n P_{L0} \tag{13}$$

The half-line equation is already known, supposing the camera had been calibrated before this step. Coming to know the laser plane parameters is also quite simple. First we measure the laser position in the world and then we get two more points, non-collinear with this one: just measure two points from the laser line when it hits any object in the world. Laser at this point is also calibrated.

Replacing t in (10) allows to get the world coordinate of any point of the image that also belongs to the line-laser. We can now jump, unequivocally, between image coordinates and world coordinates.

B. 3D Reconstruction

While the pieces are transported in the conveyor line, the camera+laser setup keeps unaltered. Taking advantage of the natural movement of the line, the laser scans every part of the objects. The analysis of successive frames, storing the 3D information extracted from the visualization of the laser lines in the images, enables us to recreate the pieces in a 3D artificial environment.

Fig. 7 presents 2 different views of the 3D reconstruction of a wavy-surface plate shown in Fig. 6, with approximately height $= 0.80m$ and width $= 0.40m$.

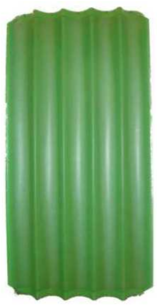

Fig. 6. Plate with an horizontal wavy surface

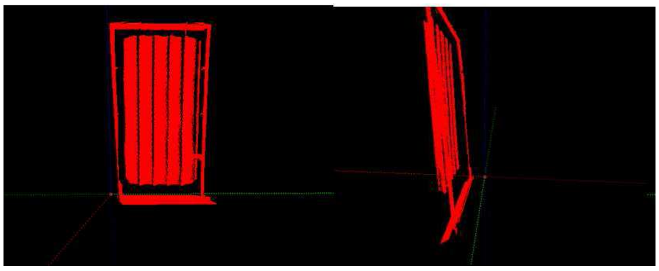

Fig. 7. 3D reconstruction of the plate of Fig. 6

Comparing the dimensions of the pieces in the 3D models with the real values, we can establish 2% for the error as an upper bound.

Visual representations give users a feel about the process of building 3D models, errors may be tracked and it enables quick validation of the system. Despite these, a more natural way to store the data was needed, other than having set of loose points in space, in order to improve upcoming algorithms and make data more organized. Hence, apart from visuals, all points were stored in a matrix form: making the indexes i and j of a matrix match with 2 of the axis of the world reference frame, and the elements associated with each pair of indexes be the coordinates on the third direction, one can map a certain volume in space into a matrix form. The points that describe the 3D models of a given object are then arranged into a data structure of this kind, following the relation shown in Fig. 8 and that can be written as:

$$\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} y_{\min} \\ z_{\min} \end{bmatrix} + \begin{bmatrix} i \\ -j \end{bmatrix} * S_c \tag{14}$$

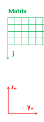

Fig. 8. Relation between matrix indexes and world reference frame

where S_c is a scale factor (relating 1 index length to its world length), subscript $'w'$ indicates world frame axis, and y_{min}, z_{min} are just the offset between matrix origin and world origin.

III. Pattern Recognition and Feature Extraction

Using the matrix representation, a segmentation algorithm was implemented so the piece shape could be evaluated. On our case study of the painting process, all objects were transported hanging from a rectangular metallic support. This evidence contributed to better segmentations of the plates and Fig. 9 shows the result of the application our algorithm to one of our 3D models.

Since the hanging object tends to deviate from standing in a perfectly vertical position, one important feature to extract is also the slope of the plate. Although other methods could be implemented eying the same goal of recognizing the objects, e.g. RFID tags, the developed vision based system allows the estimation of inclination in a trivial way and this value will show itself quite important when the painting phase initiates. Other methods would most likely need additional sensors to the same end, making the whole setup more expensive and trickier to deal with.

So, one more time, we start off with our 3D model, we adequately choose some points, and we compute the plane that best fits those points (using all of them severely slows down the computations since we are talking about almost one million points). The regression to find the plane follows a common least squares estimation, with an approach

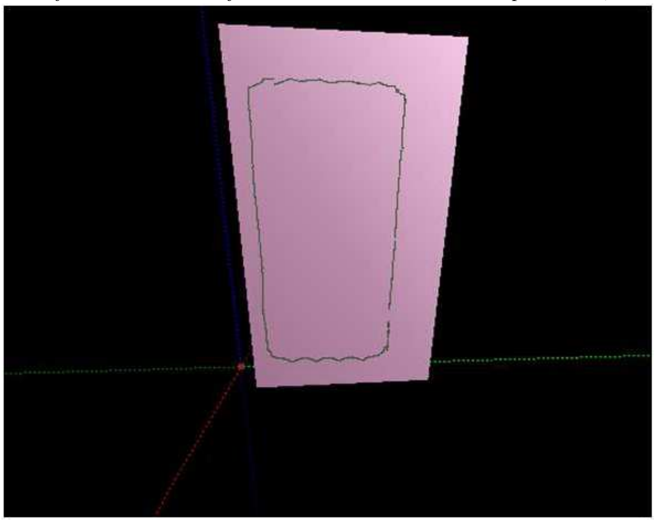

Fig. 9. Border points from the 3D model of the plate shown in Fig. 6

suggested in [4] (fundamentally, solving the least squares problem with an SVD – singular value decomposition); Fig. 10 shows a 3D simulation of a plane cutting through a model of a wavy-surface plate as the one shown previously in Fig. 6 indicating that the plane actually has the same orientation of the plate.

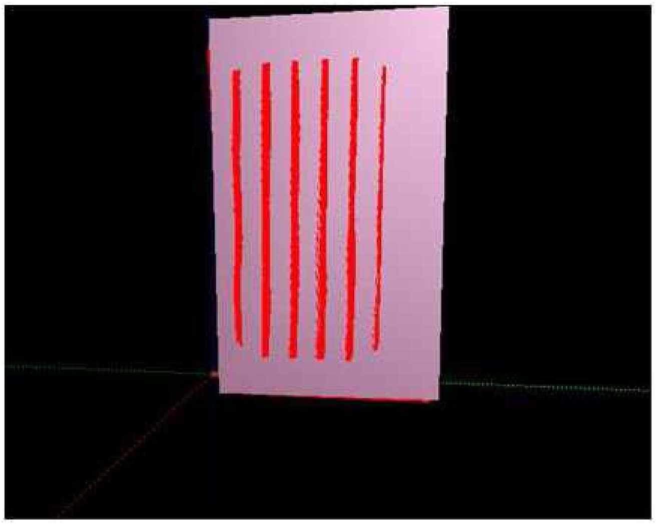

Fig. 10. Plane approximation the inclination of the plate

For the presented case study, it was required that the system could distinguish between three types of plates: one with a flat, smooth surface and two others with undulated surfaces; these two could have either a vertical wavy profile or an horizontal wavy profile.

The recognition of the shapes was carried out using a k-nearest neighbor classifier [6]. First we define N features that we must be able to extract from the 3D models. We then define a N-dimensional space, each direction corresponding to one feature. We take some pieces to paint and use them as training data, i.e., we compute the N features and them classify them manually. When running, upon detecting a new piece in the conveyor, the model is created, the features are extracted and them the distance (within the feature space) to each one of the training points is computed. Choosing the k nearest points, the new piece will be attributed the most common class among the k neighbors.

To recognize the plates, we defined the following features:

- δ^2_{mHoriz}
- δ^2_{mVert}

both of which representing the variance of depth in M slices of the 3D models, i.e., we pick three dimensional representations and slice it both horizontally and vertically as Fig. 11 suggests.

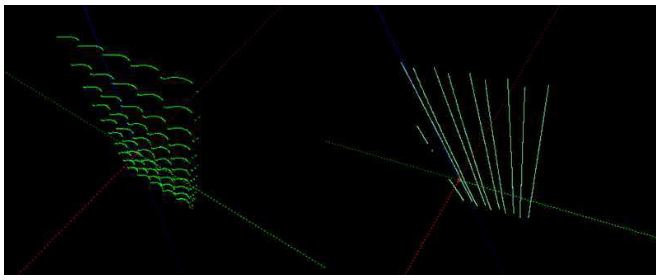

Fig. 11. Horizontal (left) and vertical (right) cuts on the 3D model shown in Fig. 7

Therefore, each δ^2_m stands for the mean of the variance on depth in those cuts, which themselves are built up from Nc points, so δ^2_m is defined as:

$$\delta^2_m = \frac{1}{M}\sum_{j=1}^{M}\left(\frac{1}{N_c}\sum_{i=1}^{N_c}\left(x^j_i - x^j_P\right)^2\right) \qquad (15)$$

It can easily be seen that, for smooth surfaces, both δ^2_{mHoriz} and δ^2_{mVert} take small values. On the other hand, a undulated surface presents high depth variance according to the direction the “waves” have, i.e., an horizontal wavy profile will have high δ^2_{mHoriz} and low δ^2_{mVert}, and vice-versa when thinking about the vertical wavy surface.

For a more generic application, where there are no obvious distinction of objects with such a few features, more complex and robust algorithms other then k-nearest neighbors may be applied, such as neural networks or support vector machines, requiring the extraction of a greater number of features perhaps, but since we have the accurate 3D model, basis are laid so that this task can be fairly easily accomplished.

Even for this application, should the need arise to distinguish more shapes, we may still maintain the used method and extract some more rich features, so this solution comes to be quite modular, flexible and expansible.

IV. Robotized Painting

One painting program to each kind of piece was developed. It was designed so that the movements are parametrized with the pieces dimensions. These programs are kept in the robots’s controller. In this case study, it is desired that the plates are painted according to their surfaces, e.g., horizontal wavy profiles should be painted with horizontal moves and never orthogonal to the waves.

Once a new object has been identified by the artificial vision system, commands are sent to the robotic manipulator (we used a MOTOMAN HP6 robot) to prepare the execution of the right program.

Before initializing the painting process, we use the object dimensions so that the program can adapt itself to the piece. This is accomplished simply by writing the values into the robot's controller, from where the developed painting programs will read the piece dimensions.

At last, another important step is to take advantage of knowing the piece inclination on the conveyor. Default programs assume the objects stand vertically; we then change the coordinate referential of the program to one that best fits the piece slop (as shown in Fig. 12).

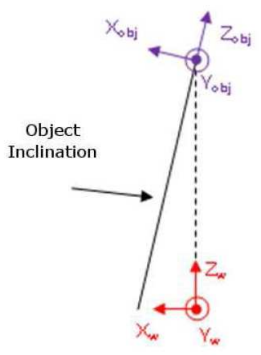

Fig. 12. World referential $OX_wY_wZ_w$ and rotated frame $OX_objY_objZ_obj$ that is aligned with the piece

So, now, the whole program will be executed according to a rotated frame that is completely described by the plane that was computed before, when analysing the 3D models.

V. Conclusions and Future Works

A. Conclusions

An adaptive scheme for an industrial painting process of small series was developed. Computer vision together with laser triangulation techniques allowed the building of accurate 3D models of objects; this subsystem may get to be useful in many other application areas,e.g., quality inspection setups.

Using the 3D model, features are extracted and a classification algorithm was implemented, that enables distinguishing between different pieces. The robotized painting cell is automatically reprogrammed using the information extracted, adapting the painting movements to the objects size and orientation.

B. Future Works

A more robust and accurate camera calibration method should be implemented for enhanced precision on scanning and, consequently, on 3D reconstructions. Classification algorithm needs further validation and test since just a few pieces were available for this work (no more then 2 of each kind).

VI. Acknowledgments

Marcos Ferreira and Paulo Malheiros acknowledge FCT for their PhD grants.

References

[1] Mark Nixon and Alberto Aguado, *Feature Extraction and Image Processing* , Elsevier, 2nd Edition, 2008.
[2] G. G. Savii, "Camera Calibration Compound Genetic Simplex Algorithm",*in Journal of Optoelectronics and Advanced Materials, Vol. 6, No. 4, p. 1255 - 1261*, 2004.
[3] Juyang Weng, Paul Cohen, and Marc Herniou,"Camera Calibration with Distortion Models and Accuracy Evaluation", *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 10*, 1992.
[4] David Eberly,*"Least Squares Fitting of Data"*, Geometric Tools LLC, 2008.
[5] ImagingSource, DMK 31BU03 camera datasheet, available on `http://www.theimagingsource.com/en_US/products/cameras/usb-ccd-mono/dmk31bu03/`.
[6] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification", *IEEE Transaction on Information Theory , 13: 21-27* , 1967
[7] David Acosta, Olmer Garca and Jorge Aponte ,"Laser Triangulation for shape acquisition in a 3D Scanner Plus Scanner", *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference (CERMA'06)* , 2006
[8] Michal Demeyere, Do Rurimunzu and Christian Eugne , "Diameter Measurement of Spherical Objects by Laser Triangulation in an Ambulatory Context",*IEEE Transactions on Instrumentation and Measurement, Vol. 56, NO. 3* , JUNE 2007
[9] Reinhard Noli and Michael Krauhausen , "Multi-beam laser triangulation for the measurement of geometric features of moving objects in production lines", IEEE 2003.
[10] Wang Lei, Bo Mei , Gao Jun and Ou ChunSheng , "A Novel Double Triangulation 3D Camera Design", *Proceedings of the 2006 IEEE International Conference on Information Acquisition, Weihai, Shandong, China* August 20 - 23, 2006
[11] Xiangyin Ma and Hongbin ,"Hybrid Scene Reconstruction by Integrating Scan Data and Stereo Image Pairs", *Sixth International Conference on 3-D Digital Imaging and Modeling* , 2007