Proceedings of the 5th WSEAS Int. Conf. on Signal Processing, Robotics and Automation, Madrid, Spain, February 15-17, 2006 (pp52-57)

Evolutionary Model-based Pose Estimation for Variable Configuration Objects

CLAUDIO ROSSI
Division of Systems Engineering and Automation
Universidad Politécnica de Madrid
Jose Gutierrez Abascal, 28006, Madrid
SPAIN

MOHAMED ABDERRAHIM,
JULIO CESAR DIAZ
Division of System Engineering and Automation
University Carlos III of Madrid
C/Butarque 15, 28111, Leganés, Madrid
SPAIN

Abstract: - The work presented in this paper concerns visual-based relative navigation of autonomous vehicles for satellite service. An evolutionary-based algorithm for the problem of pose estimation of 3D objects using 2D images is presented. The procedure consists on looking for six position parameters, three for rotation and three for translation, such that the projection of a model best fits a set of points (vertices) extracted from a 2D image. Distinguishing feature of the proposed algorithm is that the best matching model is also looked for. This feature is used for visual inspection in order to detect macro defects. A population of candidate solutions is used, whose goodness is measured in terms of distance between model and image points. Key features of the algorithm are also speed and robustness with respect to noise on the input data. Experimental results conducted both on synthetic and real images demonstrate the effectiveness of the proposed approach.

Key-Words: - Pose estimation, relative navigation, visual inspection

1 Introduction

This work presents the results of on-going research regarding visual-based relative navigation of autonomous vehicles (*chasers*) for satellite service. The task includes recognition and visual inspection of the target spacecraft, including -if possible- a first diagnosis of macro defects (e.g. a missing or undisclosed component), and a relative localization (pose) in order to plan an approaching maneuver for closer inspection. Our work is aimed at covering both aspects of the task.

Given a bi-dimensional (2D) camera image of an object, the pose estimation problem consists on finding the object's position and orientation in space (3D). Model-based methods make use of a model of the object of interest, and attempt to match some of its key features such as edges, marks, vertices, etc. to the contents of the image. A common approach is to divide the search in two steps: first, features are extracted from the image and second, a matching is looked for. Key factor of a pose estimation algorithm are robustness with respect to noise in the input data and -according to the application- speed. Input data are often noisy: extracted features can be missing or false, and their position may be inaccurate due to poor image quality, bad light conditions, partial occlusions and/or precision of the acquisition and feature-extraction process. In addition, it is worth noting that the pose problem actually consists of two sub problems: finding the position and orientation of the object and finding the correspondence between features. The *pose* (position and orientation) problem implies finding the rotation and translation of the object with respect to the camera coordinate system, while the correspondence problem consists of establishing matching image features and model features (points). The problem is difficult because it requires solution of two coupled problems, correspondence and pose, each easy to solve only if the other has been solved first. Given the matching between model and image features, one can determine the pose that best aligns those matches. If the object pose is known, one can easily determine such matches projecting the model with known pose into the original image. Most of the work presented in literature had addressed one of the discussed sub-problems and recently there is clear tendency to address them simultaneously as they appear naturally in real problems. The problem of pose estimation is central in computer vision, and has been approached with a variety of methods, like Neural Networks [15], linear programming [11]. For a comprehensive survey of different techniques of 3D object modeling, correspondence and pose estimation [5], [10] and [7] are recommended. One of the most complete algorithms that deal with the correspondence and pose estimation simultaneously is SoftPOSIT [3].

Evolutionary (or Genetic) Algorithms (EAs) [8] are a class of stochastic parallel search algorithms based on the principles of Darwin's Evolution Theory. Evolutionary algorithms keep a set of individuals, called *population* where each individual encodes a candidate solution of the problem at hand, and is called *chromosome*. Each chromosome has associated a measure of its goodness with respect to the problem, which is called its *fitness*. At each step, called *generation*, a new population is generated applying the *genetic operators*: selection, crossover and mutation. Generation after generation, good solutions emerge (evolve) towards optimality. EAs have shown their power as search procedures in several applications, especially in presence of noisy input data, and when problem-specific knowledge can be formulated as a cost function. This is the case of the problem under study, since we are in presence of noisy input data (missing points, false points, and imprecision in the coordinates). We will see in the following section a formulation of the problem in form of cost function. Thus, EAs appear to be well suited to approach the problem.

Genetic and Evolutionary Algorithms are not new to pose estimation. In [14] a genetic algorithm is used in a one-step procedure to optimise a filter-based function aimed at detecting edges in the original and model image, and measuring their match. Other applications of Evolutionary and Genetic Algorithms in Computer Vision have dealt with the problem of camera calibration [9]. EAs have also been applied in image registration [2], where the objective is finding the best transformation between an input and a reference image. Image registration has wide applications in medical imaging. Pioneer work in this field is due to Fitzpatrick [6].

In this work an evolutionary-based procedure for the problem of model-based pose estimation is proposed. The procedure consists on looking for six position parameters (three for rotation angle, α,β,γ, and three for translation, T_x, T_x, T_z, see Fig. 1) such that the projection of some points of a model best fits

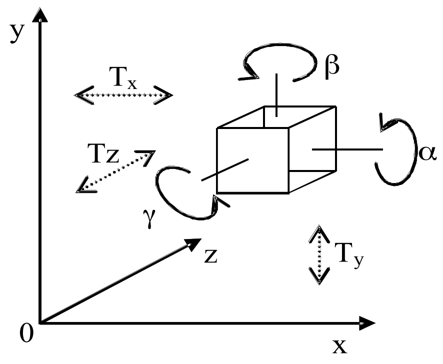

Fig. 1 Geometry of the problem

a set of points extracted from the 2D image. The vertices of the model of the object are used as reference. The search is then reduced to a 2D point-matching search. A distinguishing feature of the proposed algorithm is that it doesn't look for one object (i.e. model), but it looks between the models of different objects to match the correct one. As we will see in the following sections, this feature is used to visually detect macro defects in the spacecraft. This paper is focused on a matching procedure to search and compute the pose of an object, assuming a previous feature-extraction method has been applied to return a set of points of interest (i.e. the vertices of the object). For the purpose of assessment, we adopted a well known feature detection algorithm, SUSAN [13], and both synthetic and real images have been used. Previously published results [12] have demonstrated the speed and robustness of the pose-finding process, and its ability to effectively re-use information of a match in sequences of images, in order to track moving objects. This work is oriented towards multiple objects matching, with the final aim of implementing the algorithm on a real-time object tracking and inspection system [1] (Fig. 6).

The rest of the paper is organized as follows: next Section introduces the geometry of the problem and some notation, while Section 2 describes the proposed matching algorithm, called *EvoPose2*. Section 3 reports on the results of experiments conducted on test images. Section 4 concludes the paper with a brief discussion on open questions, which lead to future work.

1.2 Modeling of the Problem

The scene is composed by a 3D world in which an object is seen by a camera. Figure 1 shows the coordinate system of the space where the object is placed. T_x, T_y and T_z are the translation values with respect to the camera (positioned in the origin of the

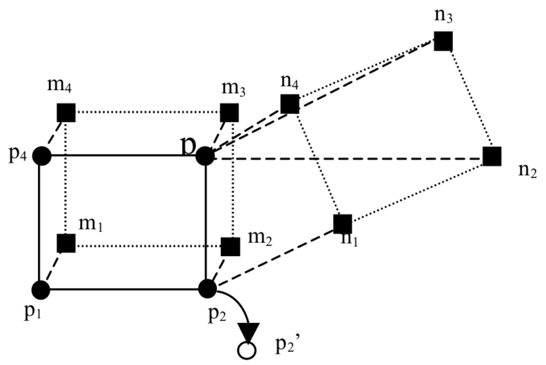

Fig. 2 Distance and matching points

axes), and α,β,γ. are the rotation angles about each axis. These six parameters are encoded in an array of floating point numbers. The precision for rotation angles has been set to 1 degree, and the precision for translation values is the maximum allowed by a 32 bit encoding of floating point numbers. A seventh parameter, the model number, is also encoded in the parameters array, which has thus 7 elements.

2 Evolutionary Pose Estimation

From the 2D view of the target object, a set of the representative points is extracted, in order to identify the object. Consider the case where the model is known. The pose estimation problem is formulated in the following way. We have to find the value of the three parameters (T_x,T_y,T_z), that give the position with reference to the camera position and distance from the camera, plus three parameters (α,β,γ) that represent the orientation of the object (see Fig. 1). If such parameters are given, then rotating and translating the model object by the corresponding values, and projecting it to the camera plane, would result in a perfect matching between camera image and model projection. In this case, the sum d of the distances between each couple of corresponding points would be zero. If the parameter corresponding, e.g., to the translation T_x is bigger, $T_x'=T_x+\Delta$, then each point of the projected model would be shifted by a quantity $f(\Delta)$ with respect to the corresponding point in the camera image. The sum of the distances in this case would be $d = n\cdot f(\Delta)$. Clearly, the closer the projection $(T_x,T_y,T_z,\alpha,\beta,\gamma)$ to the camera image, the smaller is d. Our matching problem then is turned into an optimization problem:

$$min(\ d = \Sigma_i|p_i-m_i(s)|\ ),$$

$$p_i,m_i \text{ in } R^2 \qquad (1)$$

$$s = (T_x,T_y,T_z,\alpha,\beta,\gamma)\ .$$

where p_i is a point in the image, $m_i(s)$ is the model point projected according to $s=(T_x,T_y,T_z,\alpha,\beta,\gamma)$ and | - | is the distance between two-dimensional points. Since the correspondence between points is not known in advantage, a projected model point is considered to match an image point if it is the closest. Thus, the distance measure becomes:

$$d = \Sigma_i|p_i - \min_j(\ |p_i - m_j(s)|\ )\ |\ . \qquad (2)$$

Figure 2 shows an example. Two different sets of parameters s^M and s^N lead to two different sets of projected model points, set M={m_1,m_2,m_3,m_4} and set N={n_1,n_2,n_3,n_4}. The distance between the points of set M and the image points P={p_1,p_2,p_3,p_4} is clearly smaller than the total distance of points N from points P. Therefore, the set M is a better match and the corresponding set of position parameters s^M is a better solution. The point correspondence is shown with dashed lines. In the example just described, the two set of points s^M and s^N came from the same model, projected according to different position parameters. Clearly, the same applies in the case the projected model points do not come from the same model. Hence we can have a set of position parameters applied to different models. The parameters array s is then extended with a seventh element, x, the index of the model to be projected, $s=(Tx,Ty,Tz,\alpha,\beta,\gamma,x)$, and the projected points will be the result of the application of the rotation/translation/projection process to the corresponding model's vertices.

In the proposed algorithm, the array s is a candidate solution, and a good solution is sought with an evolutionary algorithm, which keeps a population of candidate solutions and evolves them toward optimality by means of the genetic operators.

It is easy to see that if some of the points are noisy, the minimum total distance d will not be zero. This fact does not affect the search since the minimum distance in most cases still relates to a good point matching. This is shown in Figure 2, where point p_2 is moved to p_2'. Although the total distance is changed, set M still matches better than set N. The same applies when some of the points in the image are missing or are added (false points) due to poor image quality, partial occlusions and/or to faults in the segmentation process. This makes the *EvoPose2* matching algorithm very robust to noisy situations. Nevertheless, the objective function presents local minima, which means that there are combinations of parameters that lead to a small total distance, while the projected model image poorly matches the image points. This situation is especially evident in presence of highly noisy images. For this reason it is important

```
Procedure EVOLUTIONARY_ALGORITHM
BEGIN
 t := 0;
 initialize P(t);
 evaluate P(t);
 WHILE (NOT termination-condition)
DO
   BEGIN
    t := t+1;
    WHILE (|P(t)| < |P(t-1)|) DO
     BEGIN
      select parents from P(t-1);
      recombine parents
      mutate children
      evaluate children
      insert children into P(t)
     END
   END
END
```

Fig. 3 Pseudo Code of the Evolutionary Algorithm

to rely on global search methods such as EAs, where search is done sampling the solutions space.

The case of moving objects is not considered in this paper. However, it must be pointed out that for moving objects equation (2) will be dependent of time (image points p_i will change its position in each incoming image frame, becoming $p_i(t)$. Thus, the population of candidate solutions will be evaluated each time using a slightly different function, i.e. we will be using *a time varying* fitness function [12].

Finally, note that the computational effort for computing a solution only increases polynomially with the number of points under exam, which makes *EvoPose2* scale-up behavior suitable for real images, where hundreds of points may need to be considered. Figure 3 shows the pseudo-code of a typical Evolutionary Algorithm, which has been adopted for *EvoPose2*. After a set of tuning runs, the configuration described below has been adopted.

- Type: generational. All population is replaced at each generation (cf. Fig. 3). Population size: 50 individuals
- Crossover: standard single-point. Rate: 0.6
- Mutation: uniform random perturbation. Rate: 0.9; range 50% or 5% (see below).
- Selection rule: proportional roulette (each individual is selected for reproduction with a probability proportional to its fitness).
- Elitist selection mechanism, which copies the best individual of a population to the population of the next generation.
- Chromosome type: seven real valued genes.

Besides the basic EA algorithm, *EvoPose2* incorporates some heuristic rules in order to improve its performances.

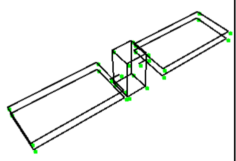	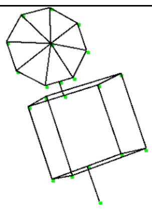	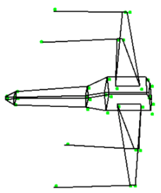
(a) Sat 1 (20x100x20)	(b) Sat 2 (50x50x135)	(c) X-Wing (190x150x60)

Fig. 4. Some of the synthetic images used to test *EvoPose* (dimensions in centimeters)

- **Tolerance**. Given a model, a value called tolerance is computed projecting the model in a way that it is centered in the camera plane, at half the maximum allowed distance. Then, it is slightly translated and rotated along the three axes, and the mean distance from the original position is added up. Tolerance represents a threshold over which we consider the mean distance is “close enough” with the aim of the tolerance parameter is to capture the notion of visual closeness of the projected model to the target image. Based on this, the algorithm terminates or takes one of the following actions.
- **Varying mutation range**. Mutation is the operator mainly devoted to exploration. At the beginning of the search the algorithm is expected to explore the whole interval, and mutation is set to vary randomly in the whole search space (range 50%). When the mean distance becomes smaller than the tolerance, it means that the algorithm is close to the solution. In this case it is desirable to reduce exploration, and focus on the exploitation of the region. Thus, perturbation range of mutation is lowered (range 5%), i.e. genes are only slightly modified, to look for better solutions in the neighborhood.
- **Kick out**. Whenever the algorithm has converged, but the mean distance is bigger than the tolerance, it is assumed that the algorithm got stuck into a local minimum. Since the population has not enough diversity, there is little hope to escape the basin of attraction of the local optimum. Then the algorithm is reset and the search is restarted from scratch.

3 Experimental Results

In order to assess the effectiveness of the proposed approach, a number of experiments have been performed. This section reports the results obtained for a subset of the images (see Fig. 4 and 5) used

Table 1. Results of experiment 1. (average over 10 runs)

Model	*SR*	*Trans (cm)*	*Rot (deg)*	*t (sec)*	*Chrom. Evals.*	*Kicks*
(a)	100%	1.26	9.40	2.12	10478	1.4
(b)	100%	1.68	3.67	1.53	10228	1.0
(c)	100%	0.76	4.73	1.01	3765	0.2

Table 2. Results of experiment 2 (average over 10 runs)

Model	*Matching model*	*Trans (cm)*	*Rot (deg)*	*t (sec)*	*Chrom. Evals.*	*Kicks*
a	a:9 b:1 c:0	1.28	12.9	2.35	5244	0.2
b	a:2 b:8 c:0	0.82	16.3	1.74	2595	1.5
c	a:1 b:1 c:8	1.28	17.3	2.18	8270	2.5

during the testing of the *EvoPose2* algorithm. All values are average over ten runs with randomly generated rotation angles and translations.

3.1 Tuning with Synthetic Images

Table 1 reports the results of the first set of experiments, aimed at tuning the algorithm. For each column, the success rate is reported along with the error in the localization (translation and rotation) and the computational efforts needed to find a solution (convergence time and number of chromosome evaluations) for the models reported in Fig. 4.

As it can be seen, the algorithm can easily find the correct model and pose. As can be expected, it is more difficult to find the correct orientation than the position. Normally, the algorithm converged very rapidly to the area where the points were located, and spent most of the time trying to find the correct orientation of the model. The correct model is also usually found in the very early stages of the search.

For what the kick outs is concerned, we noticed that most of the times they occurred when the algorithm was trying to match the wrong model. In this case of course it is extremely difficult (although not impossible) that an orientation exists such that the wrong model matches well the set of points, and after some attempts the algorithm decides that it is stuck and retry from scratch. As explained above, this decision depends on the tolerance value. Much effort was devoted to tune this heuristic rule, as it turned out to be a critical choice, in contrast with the other standard EA parameters, which settings did not appear to have a great impact on the performances.

3.2 Application to Real Images

For the second set of experiments we applied *EvoPose2* to our concrete case, the localization and inspection of satellites from B/W camera images. In order to detect macro defects from the appearance of the spacecraft we used different configurations of the model, one for each typical defect of a given satellite. According to which model best matches the image, the algorithm can tell which can be the problem. Fig. 5 shows the three configurations used: (a) normal, (b) missing (or unopened) solar array and (c) partially opened solar array, together with a picture showing the set of points fed to the algorithm and a good matching pose/model.

Using real images makes the pose problem harder, mainly because of the highly noisy information (points) provided to the algorithm. Nevertheless, *EvoPose2* has demonstrated a satisfactory behavior for what the task of recognition is concerned, although the nature of the points (many 'false positives') and the fact that the three models were similar caused the algorithm do misclassify some instances.[1] Table 2 summarizes the results. The precision of the pose could not be as good as the previous experiment, especially for what orientation is concerned.

4 Discussion and Conclusion

The application of the proposed algorithm to real images, as expected, has raised some issues, which we are currently addressing.

Although the precision of the localization can be improved by imposing stricter tolerances, we do not consider this fact very important for our purposes, since search will be refined in al later moment: *EvoPose2* works in real time (has no termination criterion) and will constantly be working as new images are provided, either because the relative position of chaser/target has changed, or because its output has been used to control the camera for a better view ("zoom and center") .

From the algorithmic point of view, we believe the precision of the pose has worsened mainly due to the poor quality of the extracted features. However, let us point out the following issues:

- The precision of the localization can be improved by imposing stricter tolerances, at cost of higher running times.
- Working in space has the advantage that the absence of atmosphere makes the image sharper, a fact that benefit the feature extraction process.
- Many non-corner points were present: it will be important in the future to use this information as feedback to construct the model according to output of the feature extraction process, including all the points that appear typically.

[1] When the algorithm could not converge in less than 10 seconds the instance was considered misclassified. In this case only the current model was returned as answer, discarding the pose

The presence of poor local minima can result in misclassifications and bad localization. This can be due to the use of the mean distance alone as optimization function. A way to overcome this problem would be to incorporate other problem-specific information such as color and light information. We are currently working on this aspect. Also, the integration with other sensing devices (e.g. laser for distance), would help the search.

The improved version of the proposed algorithm will be used in a 3D object tracking application developed in our Laboratory (see Fig. 6) and could be applied to other vision systems in robotic applications such as visual servoing and vision based navigation.

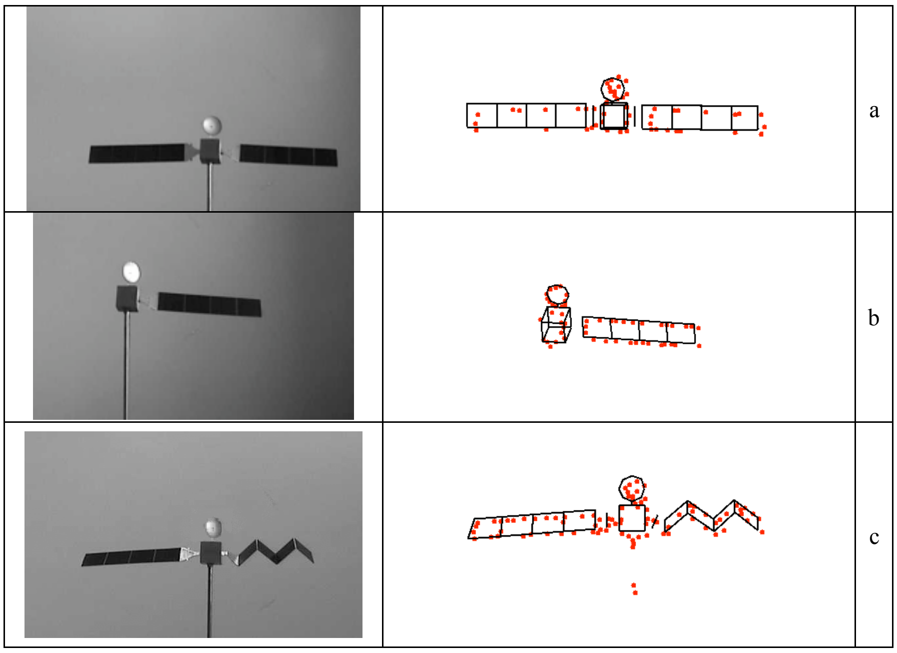

Fig. 5. An application to real images for defects detection (model dimension: 38x3x3.7 cm)

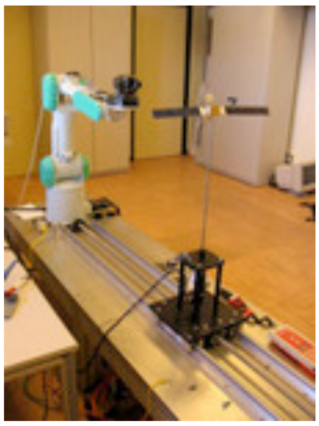

Fig. 6. Test Platform

References:

[1] M. Abderrahim, *et al.*, Mechatronics Testbed for Vision Based Navigation. *Proc. of 9th Mechatronics Forum International Conference,* 2004, pp. 415-423.

[2] L.G. Brown, A Survey of Image Registration Techniques, *ACM Computing Surveys*, Vol.24, No.4, 1992, pp. 325-376.

[3] L.S. Davids, *et al.*, SoftPOSIT: Simultaneous Pose and Correspondence Determination, *Proc. 7th ECCV*, Vol. III, 2002, pp. 698-703.

[5] O. Faugeras, *Three-Dimensional Computer Vision – A Geometric Viewpoint*, MIT Press, Boston, MA, 1993.

[6] J.M. Fitzpatrick, J.J Grefenstette, D. Van Gucht, Image Registration by Genetic Search. *Proc. of IEEE Southeastcon*, 1984, pp. 460-464

[7] S.E. Gold, *et al.*, New Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence. *Pattern Recognition*, Vol.3:8, 1998, pp. 1019-1031.

[8] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning* Addison-Wesley, New York, 1989.

[9] S. Hati, S. Sengupta, Robust camera parameter estimation using genetic algorithm, *Pattern Recognition Letters*, No.22, 2001, pp. 289-298.

[10] R. Horaud, F. Dornaika, B. Lamiroy, S. Christy, Object pose: the link between weak perspective, paraperspective, and full perspective, *International Journal of Computer Vision*, Vol:2, 1997, pp. 173-189.

[11] D.G. Lowe, Fitting Parameterized Three-Dimensional Models to Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.13, No.5, 1991, pp. 441-450.

[12] C. Rossi, M. Abderrahim, J.C. Díaz, An Evolutionary Algorithm for Model-based Pose Estimation and Tracking, *Proc. Visual Information Engineering,* 2005, pp. 227-234.

[13] S.M. Smith, J.M. Brady, SUSAN - a new approach to low level image processing, *International Journal of Computer Vision*, Vol.23, No.1, 1987, pp. 45-78.

[14] F. Toyama, K. Shoji, J. Miyamochi, Model-based pose estimation using genetic algorithm, *International Conference on Pattern Recognition*, 1998, pp.198-201.

[15] P. Wunsch, G. Hirzinger, Registration of CAD-models to images by iterative inverse perspective matching, *Proc. Int. Conf. Pattern Recognition,* 1996, pp. 78-83.