IEICE Electronics Express, Vol.7, No.14, 969–975

Autotuned feature detection to improve position based visual servoing

Khalid Mahmood[a)] **and Naeem Iqbal**[b)]

Department of Electrical Engineering

Pakistan Institute of Engineering and Applied Sciences, Islamabad, Pakistan

a) *phd0506@pieas.edu.pk, email2khalid@yahoo.com*

b) *naeem@pieas.edu.pk*

Abstract: This paper tackles challenges of general realtime position based visual servoing without using model information. These are mainly speed limitations of suitably affordable vision processing system, generally reliable features and faithfully tracking the target to keep it in field of view. Feature detection is not only time consuming but also depends on actual feature parameters. The proposed autotuned feature detection enables the system in reaching a higher speed by reducing processing time and also has the advantage of easiest portability to another vision processing system. Simulations with real images and experiments have been conducted to prove the effectiveness of the method.

Keywords: position based visual servoing, autotuned feature detection, keeping target in field of view, general visual servoing

Classification: Science and engineering for electronics

References

[1] M. Saedan and M. H. Ang. Jr. H, "3D vision-based control on an industrial robot," *Proc. IASTED Int. Conf. on Robotics and Applications*, Florida, USA, pp. 152–157, 19-22, Nov. 2001.

[2] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," *The confluence of vision and control*, vol. 237 of LNCIS Series, Springer Verlag, pp. 66–78, 1998.

[3] G. Chesi, H. Koichi, D. Prattichizzo, and A. Vicino, "Keeping features in field of view in eye-in-hand visual servoing: A switching approach," *IEEE Trans. Robot. Autom.*, vol. 5, no. 20, pp. 908–913, Oct. 2004.

[4] C. Harris and M. J. Stephens, "A combined corner and edge detector," *Alvey Vision Conference*, pp. 147–152, 1988.

[5] O. Faugeras and Q. T. Long, *The geometry of multiple images*, Cambridge, MA, MIT Press, 2001.

[6] M. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Commun. ACM*, vol. 6, pp. 381–395, 1981.

[7] G. Schweighofer and Pinz, "Global optimal o(n) solution to the pnp problem for general camera models," *Proc. BMVC*, 2008.

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010

[8] E. Andreas, A. Neubeck, and L. V. Gool, "Generalised Linear Pose Estimation," *British Machine Vision Conference (BMVC)*, Sept. 2007.

1 Introduction

Model based visual servoing systems are fast [1] as target geometry is rigid, well known and prominently marked features with known locations are used. This severely restricts visual servoing where we cannot use highly prominent marks on objects. In some cases of Position Based Visual Servoing (PBVS) and Image Based Visual Servoing (IBVS) [2], severe problems do arrive for example in PBVS the target may get out of the field of view of camera (FOV) because of processing delay and also in highly noisy environments [3] which leads to failure. To overcome the problem proposed autotuning equations greatly reduce processing delay. Because corners are most reliable features which can be detected even when lines cannot be detected (especially when lines are parallel to direction of motion) and corners are also invariant to rotation, translation (and scaling to large extent). It is quite challenging to formulate a general approach for visual servoing which does not depend on prior model information, and is also self-tuning according to the vision processing system. A solution to the problem demands that the vision processing system must automatically adjust for lesser accuracy if the relative speed goes higher, so that it comes out of fine calculations and gives its best to keep the object in camera view. As the speed decreases, the system should provide more accurate results. If the vision algorithm is implemented on another platform, it should adjust maximum allowed relative speed and maximum tolerable error, at higher speeds, to as low as possible. Basic idea of the research presented in this paper is auto-tuning equations for general purpose position based visual servoing. It can be used with classifier which after recognition windows the object regions and calls feature-extractor within the window (windows). It can also be used where prominently marked features are used where feature occlusion may occur and number of identified features (e.g. less than 4) can not determine pose, the proposed method finds redundant corners within the window of identified features to determine pose.

2 Harris corner detector

Harris detector [4] becomes a highly reliable, robust and fast if used with precalculated template of Gaussian circular function which gives optimal smoothing to reduce noise and preserves image features. It is based on:

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \tag{1}$$

where $A = I_x^2 \otimes w$, $B = I_y^2 \otimes w$, $C = (I_x I_y)^2 \otimes w$, I_x and I_y are image gradients along x and y direction respectively, $\otimes$ is convolution operator and w is the template of Gaussian smoothing filter which for row i and column j

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010

of the image can be, $w_{i,j} = \exp \frac{-(i-x)^2-(j-y)^2}{2\times\sigma}$). For metric matrix R:

$$R = AB - C^2 - k(A+B)^2 \tag{2}$$

Where k is sensitivity parameter, with smaller value of k, likelihood of sharp corners detection inreases. In eq. (2)$R = R(x, y)$, and it is positive in the corner region, negative in the edge region and small in flat region. Now, corners belong to maxima of R for $R \geq 0$ which is ensured by $0 \leq k \leq 0.25$. We can set a threshold R_{th} such that a corner is detected for $R \geq R_{th}$. Using eq. (2) directly avoids calculating eigenvalues which gives processing speed and adds flexibility of using k. The number N of corners detected increases with lower value of k. which beyond a certain maximum, severely limits the maximum attainable speed of control. Again for robustness, redundancy is required, and N must not fall below a minimum value. It was practically found under various conditions that $0.5 \leq \sigma \leq 1.82$ is most reliable range which also allows us to use auto-tuning of k. For other ranges of σ, k has lesser effect. $\sigma = 0.97$ gives optimal results in most cases (noise level$\leq$ 10 pixels). If noise level increases then σ should be increased for more smoothing. Weak corners should be suppressed with non-maximum suppression.

3 Position Based Visual Servoing (PBVS)

The main difficulties with PBVS are related to the difficulty of building 3D representation of the target in real time. Because this approach has no direct control over the image itself, and processing delay can cause the object of interest to leave the FOV [3] if image acquisition, processing, pose estimation and control are not tuned to the situation. Therefore the proposed approach is aimed to improve PBVS by auto-tuned feature detection and robust feature selection (like maximum likelihood/RANSAC [6], etc.) under the limitations of processing system. Note that techniques of limiting the search criterion for point (or corner) matching fail under various conditions, especially at higher speeds when high percentage of features occlude and appear. For example, if there is no translation (pure rotation) between two successive frames, then epipolar geometry fails and we cannot limit search to epipolar line. It also fails badly at higher speeds when many features occlude and appear. Moreover epipolar geometry is not known before features are matched. So, for a general case, it is recommended to use robust methods, like RANSAC [6]. Remarkable improvements can be achieved by using hardware implemented SAD (sum of absolute difference) algorithm for feature matching.

3.1 Autotuned position based visual servoing

Successive image blur or distortion increases as camera velocity increases, or feature occlusion may increase, which in turn results in lower number of detected corners. This either decreases redundancy and robustness (or results in failure of control). There may also be a large no of detected features if a larger number of occluded features suddenly appear again. This decreases feature processing speed and limits maximum attainable speed of the system.

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010

To overcome all such problems, we have to optimize feature detection such that it can be detected when it is faint or strong, and we also have to set an optimized upper limit on the number of features selected for processing, depending on the speed benchmark index of the hardware+algorithm.

Let I_1 and I_2 be the previous and current image respectively, $L_1 = [L_1x\ L_1y]^T$, $L_2 = [L_2x\ L_2y]^T$ be the locations of features (e.g., corners) in the respective images. Let there be n_1 features in I_1 and n_2 features in I_2, and $n = min(n_1, n_2)$, then the steps are:

1. Find matching points by using maximum likelihood or least sum of absolute difference, SAD, of square window of size w (e.g., w=4) around the features of images. Let there be SAD value $S_{i,j}$ for comparing ith feature in I_1 with jth feature in I_2 given by:

$$S_{i,j} = \sum_{x=-w}^{w} \sum_{y=-w}^{w} |I_1(L_{ix} + x, L_{iy} + y) - I_2(L_{jx} + x, L_{jy} + y)| \quad (3)$$

$$s_k = min(S_{i,1}, S_{i,2}, S_{i,3}, ..., S_{i,n2}), i = 1..n \quad (4)$$

$$or\ s_k = min(S_{1,j}, S_{2,j}, S_{3,j}, ..., S_{n1,j}), j = 1..n \quad (5)$$

where n is minimum number of features detected in either image. Use eq. (4) if n=n1, otherwise use eq. (5). Now if s_k is arranged in ascending order, then first n feature pairs related to s_k are matched.

2. Use reliable methods like RANSAC to verify that features have been correctly identified and mapped. Also let μ=$\frac{1}{n}$ $\sum_{k=1}^{n}$ s_k.

3. Find the geometrical transformation that best describes the mapped features. This can be implemented as a fitting function in RANSAC. Alternatively other methods can be used to find rotation R and normalized translation $T_n = T/||T||$ of the camera displacement [5].

4. Calculate or chose the camera velocity $V = [v\ w]^T$ with PID controller. Any other control can be used but for simplicity consider the control scheme used by [3] where camera velocity is chosen such that object remains in FOV. The calculations are based on the following equation:

$$\begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} \lambda_i \mu T_n \\ \lambda_r R \end{bmatrix} \quad (6)$$

5. Tune the feature detection parameters k, R_{th} and N by: $\delta k \propto V$, $\delta N \propto -\frac{V}{i_s}$, $\delta\ R_{th} \propto -\frac{V}{i_s}$. And in terms of iterative calculations:

$$k_i = k_1 + c_1|v_i + w_i| \quad (7)$$

$$N_i = N_1 - c_2\frac{|v_i + w_i|}{i_s} \quad (8)$$

$$R_{thi} = R_{th1} - c_3 m\frac{|v_i + w_i|}{i_s} \quad (9)$$

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010

The second term in each equation represents update. The constants c_1, c_2 and c_3 are defined such that k, R_{th} and N remain within bounds with the desired lower and upper bounds of the camera velocity and m is the mean of the local image window in corner detection. The initial values of k, R_{th} and N are k_1, R_{th1} and N_1. Examples at the end show the values of these constants taken and results have been plotted.

4 Results and comparisons

Experimental setup was based on TMS320DM642AZD board with 720 MHz fixed point DSP processor, XDS 560JTAG, code composer studio v3.2, CCD color camera LTC 450/50 and 6DOF Robot RV12-SLC. The statistics and calculated control commands were sent to Intel's dual core processor 2.4 GHz based PC running windows XP SP3. The settings were: Speed index i_s is set to 1 for the board. This can be set to any value but it will inversely change the values of c_1 and c_2 on the same platform. These constants remain unchanged if i_s is changed proportional to the time profile of algorithm on another (e.g. better/faster) system. This has been verified by profiles of the same algorithm coded in Matlab on different computers and shows easiest portability. For better results, the recommended bounds of k, R_{th} and N are: $0 \leq k \leq 0.25$, $0.00001 \leq R_{th} \leq 0.001$, and $10 \leq N \leq 300$. And initial values being $k_1 = 0.12, R_{th1} = 0.0001$ and $N_1 = 300$. In our case maximum allowed speed was ± 5 m/s for translational component and ± 0.5 rad/s for rotational component of camera velocity in eye-in-hand configuration. Values of the constants were $c1 = (0.24 - k1)/ceil(5 + .5) = 0.02$, $c2 = (N1 - 10)/ceil(5 + .5) = 48.333$ and $c3 = Rth1/(ceil(5 + .5) + 1) = 1.4286e - 5$. Results of auto-tuning are shown in following figures. Note 1 iteration=28.073 ms. Fig. 1 and fig. 2 show changes in parameters k,N and R_{th} as velocity changes. Weak corners are

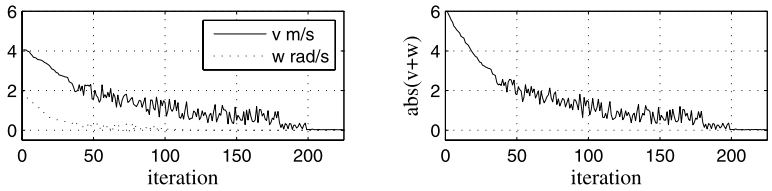

Fig. 1. Velocity components and absolute velocity

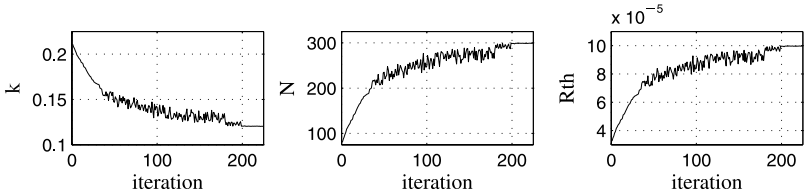

Fig. 2. Auto-tuned values of k,N and R_{th}.

suppressed along with noise by non-maximum suppression. Fig. 3 shows the effect of σ, non-maximum suppression radius r and sensitivity parameter k on rotational and translation error. Note that with higher σ and r noise is suppressed along with weak corners which improves accuracy but speed of

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010

processing is reduced as non-maximum suppression radius is increased. To increase speed by 25 percent, k was increased and $r = 2$ set as shown in fig. 3 part 3. This resulted in about same error level as in part 1 of fig. 3 but with 25 percent lesser CPU time. In all other simulations and experiments $r = 2$ was kept fixed and k was varied relative to the speed of object w.r.t camera. Autotuned method proved far efficient than other general methods based on

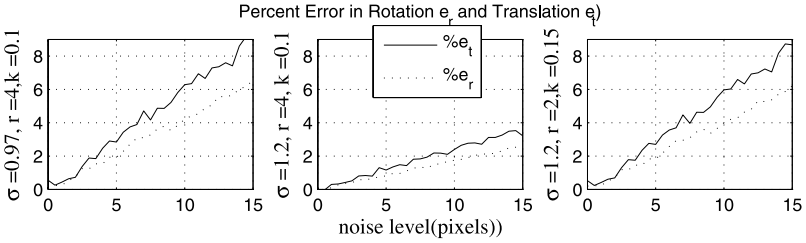

Fig. 3. Percent Errors in Rotation and Translation.

point features. Although PnP methods are even faster than those which do not depend on known prior correspondence between image points and world frame. A comparison, fig. 4 part 1, of the autotuned method with Globally Optimal Pose and SeDuMiSolver [7] for rotational and translation error less than 6 percent shows improved response of the proposed method. This is because the proposed method uses variable points within limits of maximum N and minimum 5 point, it takes lesser CPU time and tracks the target faster using maximum N points compared to algorithms which use fixed N. Fig. 4 part 2 and 3, show the higher accuracy of the method with same CPU time and variable N ($5 \leq N \leq 300$) against Generelized Linear Pose Estimation (GLPE) and Fiore methods [8] which use fixed N.

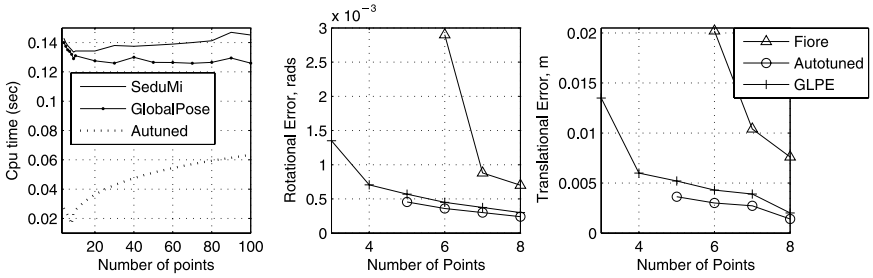

Fig. 4. CPU Time (part 1) and Error (part 2,3) comparison for N points.

5 Conclusion

The tests were successful at 30 frames/sec with any of TCP/IP, UDP and HRTDX connections used for sending commands and results. As abs $(v + w)$ increases, k increases reducing n_1 and n_2 (number of features to be matched). R_{th} decreases to counter the effect of blur. Blur effect is also countered by increased value of k. Also N (the number of feature pairs to be used for pose

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010

determination) decreases reducing processing time. In this way auto-tuned feature detection greatly resolves the speed and portability issues of real-time visual servoing. Note that with Harris detector, edges can be detected for $R \leq 0$ using eq. (2). So, the algorithm can be easily adopted for edges as well. In future this method will be used for corners and edges, with a classifier to recognize objects, window the region of object, use autotuned method within the window and find pose/track in an artificially intelligent environment.

DOI: 10.1587/elex.7.969
Received January 15, 2010
Accepted June 14, 2010
Published July 25, 2010