Multi-View Visual Servoing using Epipoles

Jacopo Piazzi
Johns Hopkins University
Baltimore, Maryland USA

Noah J. Cowan
Johns Hopkins University
Baltimore, Maryland USA

***Abstract*— We explore the benefits of multiple views for visual servoing (VS) by commanding a single camera to first "peer" at a scene from two vantage points, thus acquiring a set "reference" images, prior to executing a visual position task. Our approach completely decouples the translational and rotational components of our controller: epipoles from the reference views drive the translational error to zero, while the rotational degrees of freedom maintain all of the features in the field of view (FOV). We furnish a simple Lyapunov stability proof that demonstrates a large domain of attraction while maintaining all features in the FOV. Finally, we present simulated experiments that suggest robustness to measurement noise and large variations in the baseline between the reference views.**

I. Introduction

Insects often exploit parallax rather than stereo to perceive depth [1]. We explore the benefits of multiple views for visual servoing (VS) by commanding a single camera to first "peer" at a scene from two vantage points, thus acquiring a set "reference" images at the beginning of a visual position task. In many visual control strategies only the desired and the current views are employed to accomplish the visual servoing task. Once a new reference image is acquired the old one is discarded. In this paper we propose to employ two previous images in order to perform a more robust and stable visual control task. The VS algorithm presented drives the epipoles between the reference views and the current and desired views into congruence.

Our approach completely decouples the translational and rotational components of our controller: epipoles from the reference views drive the translational error to zero, while the rotational degrees of freedom maintain all of the features in the field of view (FOV). Interestingly, the translational control problem reduces to stereo control of a single point feature: the epipoles serve as a pseudo-projection of the optical center of the camera to the reference image planes treated as a stereo rig. Our stereo controller closely follows that of Hager et al. [2], and we furnish a simple Lyapunov stability proof that demonstrates a large domain of attraction. In addition our method guarantees the maintaining of all features within the FOV.

Epipolar geometry has also been exploited in prior works on VS: Rives describes an algorithm that drives robot to a desired pose by steering the image features along epipolar lines [3]. Previous VS algorithms based on epipolar geometry recover the rotational misalignment by decomposing the essential or the homography matrix [4]–[6]. Our method, which does not rely on matrix decomposition is convergent and (at least empirically) robust to image noise and errors in the baseline between the two reference views. We note that Malis *et al.* [5] and Taylor *et al.* [6] prove the robustness of their algorithms with respect to intrinsic and extrinsic parameters.

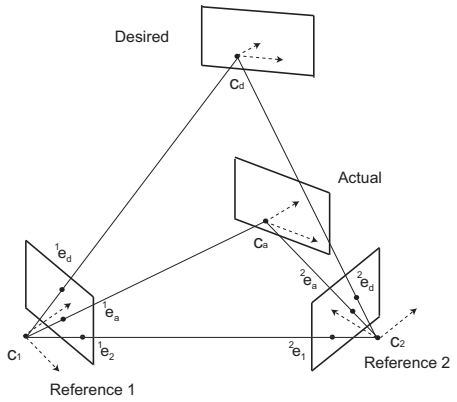


Fig. 1. During a "training" phase, a fully actuated camera acquires a desired image from an unknown position. Then, the camera acquires two additional reference images from two distinct vantage points. The VS algorithm then attempts to move the camera's actual position at each step so as to drive the epipoles from the reference images into congruence with the desired epipoles, while maintaining the features in the FOV with the rotational DOFs.

Our algorithm requires modest computation, namely the estimates of two essential matrices (via the linear 8 point algorithm), and does not require subsequent decomposition of those matrices. Moreover, we need not compute the Jacobian "online"; following Kim *et al.* [7], we employ a precomputed Jacobian at the goal location and nevertheless guarantee convergence. We present simulation experiments to validate the method.

II. Theoretical background

A. Camera model

Let ${}^ip \in \mathbb{R}^3$ denote the location of a point, p, written with respect to a rigid frame $\mathcal{F}_i$. Let ${}^ih_j = ({}^iR_j, {}^iT_j) \in \mathrm{SE}(3)$ denote the transformation between rigid frames $\mathcal{F}_i$ and $\mathcal{F}_j$, i.e. ${}^jp = {}^jh_i({}^ip) = {}^jR_i{}^ip + {}^jT_i$ is the location of the point p with respect to $\mathcal{F}_j$.

Consider a perspective projection camera, and attach a rigid frame, $\mathcal{F}_a$, to the camera, with the frame origin coincident with the optical center, c_a, and Z-axis aligned along the optical axis. Let $\mathcal{F}_w$ be a stationary world reference frame, and ${}^wp$ be a point expressed in that frame. Then the projection, ${}^ax$, of ${}^wp$ to the image plane is given

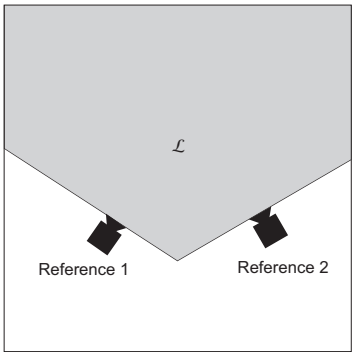


Fig. 2. $\mathcal{L} \subset \mathbb{R}^3$ contains all points in front of both reference cameras.

by

$$^a x = \pi \circ {}^a h_w({}^w p) \tag{1}$$

where "$\circ$" denotes function composition and $\pi : \mathbb{R}^2 \times \mathbb{R}^+ \to \mathbb{R}^2$ is given by

$$\pi(p) := \frac{f}{p_3} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix}, \quad p_3 > 0 \tag{2}$$

with f camera's focal length (for clarity of presentation, we disregard other intrinsic camera parameters).

Most eye-in-hand VS algorithms require a "training" stage: a user commands a robot with attached camera to move to a desired location, $\mathcal{F}_d$, capture an image, and store the image-plane location of a set of feature points in the scene. The VS algorithm presented in this paper requires two additional "reference" images taken from two distinct locations in space, $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively. During the course of servoing, the actual camera, located at $\mathcal{F}_a$, moves, while the reference and goal frames remain fixed in space.

We treat the reference cameras as a stereo pair. The projection of a point to a stereo pair of cameras is given by the mapping

$$g : \mathcal{L} \mapsto \mathbb{R}^4 \tag{3}$$

with

$$g({}^w p) = \begin{bmatrix} \pi \circ {}^1 h_w({}^w p) \\ \pi \circ {}^2 h_w({}^w p) \end{bmatrix}, \tag{4}$$

where π is given in (2) and $\mathcal{L} \subset \mathbb{R}^3$ is the set of all the points in front of both cameras, as shown in Fig. 2.

B. Triangulation problem

Knowledge of the cameras' relative position and orientation together with knowledge of the internal camera parameters permits the selection of a "pseudo-inverse" or "triangulation function" [8],

$$g^\dagger : \mathbb{R}^4 \mapsto \mathcal{L}. \tag{5}$$

Roughly speaking, the triangulation function, $g^\dagger$, triangulates a 3D point from the intersection of the two corresponding rays in space. Note that 3D points on the baseline all project to the same image tuple, and thus $g^\dagger$ does not provide a unique solution baseline points. Thus we define the subset $\mathcal{W} \subset \mathbb{R}^3$ where g is invertible, namely

$$\mathcal{W} = \mathcal{L} - \mathcal{B} \tag{6}$$

where the thin set $\mathcal{B}$ is the baseline joining c_1 and c_2. We now have that $g^\dagger \circ g$ is the identity mapping on $\mathcal{W}$.

C. Epipolar geometry

We briefly review epipolar geometry in order to fix the notation used in subsequent sections; for a more through treatment see [9], [10]. Given a pair of cameras, the *baseline*, or line joining the two cameras' optical centers, intersects each image plane at the *epipoles*. Each of the four cameras $\{c_a, c_d, c_1, c_2\}$ possesses three epipoles, one for each of the other cameras (Fig. 1). We denote by ${}^i e_j$ the epipole between camera i and j as seen by camera i or, in other words, the projection to camera i of camera center j.

Consider a set of feature points p_k, $k = 1, \ldots, N$, and denote their respective images in camera i by

$$^i x_k = \pi \circ {}^i h_w({}^w p).$$

For any pair of cameras, i and j, the *essential matrix* $E_{ij} \in \mathbb{R}^{3\times 3}$ [9], satisfies

$$\begin{bmatrix} ^i x_k \\ 1 \end{bmatrix}^T E_{ij} \begin{bmatrix} ^j x_k \\ 1 \end{bmatrix} = 0. \tag{7}$$

In general the essential matrix has rank 2 and is defined up to an arbitrary scale. The (one dimensional) right null space of E_{ij} represents the epipole ${}^j e_i$ on the image plane in homogeneous coordinates. The essential matrix depends on the relative position of two views and can be expressed as

$$E_{ij} = \widehat{{}^i T_j}\, {}^i R_j, \tag{8}$$

where $\widehat{\ }: \mathbb{R}^3 \to \mathfrak{so}(3)$, i.e. $\hat{T}$ is the usual skew symmetric matrix associated with $T \in \mathbb{R}^3$. Assuming there are at least eight corresponding points in general configuration (not belonging to a critical surface), we can estimate the epipolar geometry from the images by exploiting for example the 8-point algorithm [11] and thus obtain the epipoles ${}^1 e_a$, ${}^1 e_d$ and ${}^2 e_a$, ${}^2 e_d$, which are the reference camera projections of c_a and c_d, respectively. In other words, we can write

$$e_a = g(c_a) \quad \text{and} \quad e_d = g(c_d) \tag{9}$$

where $e_a = [\ {}^1 e_a,\ {}^2 e_a\ ]^T$ and $e_d = [\ {}^1 e_d,\ {}^2 e_d\ ]^T$. Thus, using the two reference views, we can triangulate the 3D location of the camera centers, given the epipoles, namely

$$c_a = g^\dagger(e_a) \quad \text{and} \quad c_d = g^\dagger(e_d). \tag{10}$$

III. Controller design

We suggest two decoupled controllers, one for steering the actual camera on the target and the other one for maintaining the scene in the FOV and align the camera orientation to the target. In particular we exploit the epipoles from the reference views to drive the translational error to zero with global convergence. We employ the remaining rotational degrees of freedom to maintain all of the features visible.

Every time one camera undergoes a change of position, the points on the image plane, change their locations. The relationship between point velocities and the camera velocity is expressed by the image Jacobian. Let p represent a point in space and $\dot{p}$ its velocity. Let x represent the projection of the point to the image plane and $\dot{x}$ the corresponding velocity. The Jacobian that maps Cartesian velocities (written in the camera frame) to image-plane velocities is given by

$$\dot{x} = D\pi(p)\dot{p}, \tag{11}$$

where

$$D\pi(p) = \frac{1}{p_3}\begin{bmatrix} 1 & 0 & -\frac{p_1}{p_3} \\ 0 & 1 & -\frac{p_2}{p_3} \end{bmatrix}. \tag{12}$$

In a stereo camera configuration a Cartesian velocity in the world frame, ${}^{w}\dot{p}$ induces image-plane velocities in both cameras, namely

$$\begin{bmatrix} {}^{1}\dot{x} \\ {}^{2}\dot{x} \end{bmatrix} = J({}^{w}p)\,{}^{w}\dot{p} \tag{13}$$

where the Jacobian associated with the point ${}^{w}p$ is given by

$$J({}^{w}p) = \begin{bmatrix} D\pi({}^{1}p)\,{}^{1}R_w \\ D\pi({}^{2}p)\,{}^{2}R_w \end{bmatrix} \in \mathbb{R}^{4\times 3}$$

with ${}^{i}R_w$ the rotational matrix between camera i and world frame.

A. Translational controller

The translational control objective is to steer the point c_a to the desired point c_d using only the epipoles. We interpret the epipoles as ordinary feature points on the reference images, namely $e_a = g(c_a)$ and $e_d = g(c_d)$. Of course the epipoles are not directly "visible"; they are computed from the essential matrix as described above.

Our controller employs the following interesting property of a stereo system [8], that we apply to the epipoles:

$$e_d - e_a = \Gamma(c_a)\Gamma^{-1}(c_d)J(c_a)(c_d - c_a) \tag{14}$$

$$= \Gamma(c_d)\Gamma^{-1}(c_a)J(c_d)(c_d - c_a) \tag{15}$$

where

$$e_d = \begin{bmatrix} {}^{1}e_d \\ {}^{2}e_d \end{bmatrix}, \quad e_a = \begin{bmatrix} {}^{1}e_a \\ {}^{2}e_a \end{bmatrix}$$

and J denotes the Jacobian of the stereo perspective transformation (13). The 4×4 matrix Γ is given by

$$\Gamma(c_a) = \begin{bmatrix} (\Pi_3\,{}^{1}T_a)I_{(2\times 2)} & \mathbf{0} \\ \mathbf{0} & (\Pi_3\,{}^{2}T_a)I_{(2\times 2)} \end{bmatrix} \tag{16}$$

where ${}^{1}T_a$ and ${}^{2}T_a$ denote the vectors from cameras 1 and 2 to the point c_a and $\Pi_3 = [0, 0, 1]$. See Fig. 3.

For clarity of presentation we drop parenthesis for the entities in (14) and (15), namely $\Gamma(c_a) = \Gamma_a$ and $J(c_a) = J_a$. Let $\tilde{e} = e_d - e_a$ and $\tilde{c} = c_d - c_a$, so that (15) becomes:

$$\tilde{e} = \Gamma_d\Gamma_a^{-1}J_d\tilde{c} \tag{17}$$

Note that Γ_a and Γ_d^{-1} are matrices positive definite as long as the points c_a and c_d belong to $\mathcal{L}$.

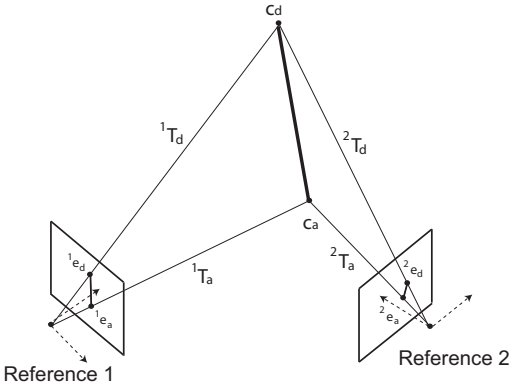


Fig. 3. Stereo system given by the reference cameras 1 and 2. There is a simple mapping (14) between the Cartesian error $c_d - c_a$ and the image plane error $e_d - e_a$.

Through the knowledge of the epipoles it is possible at every instant to compute the spatial displacement of c_a. One may employ this information to design a control scheme according the Newton–Raphson approach like

$$\mathbf{u} = J_a^{\dagger}\tilde{e} \tag{18}$$

where $J_a^{\dagger}$ is the Moore-Penrose pseudo-inverse, namely $J_a^{\dagger} = (J_a^T J_a)^{-1}J_a^T$. The controller in (18) guarantees convergence arising from traditional and well known analysis. However, as pointed in [7], this strategy seems to render the algorithm very sensitive to calibration and sensor error as well as to the epipole estimation.

To avoid these problems we adopt the same strategy as in [7], where the controller uses fixed gain laws based only on the desired set point. Therefore we denote the Γ_d-weighted pseudo inverse as

$$J_d^{\dagger} := (J_d^T\Gamma_d^2 J_d)^{-1}J_d^T\Gamma_d^2 \tag{19}$$

and the Newton type controller such that

$$\mathbf{u} = J_d^{\dagger}\Gamma_d^{-1}\tilde{e}. \tag{20}$$

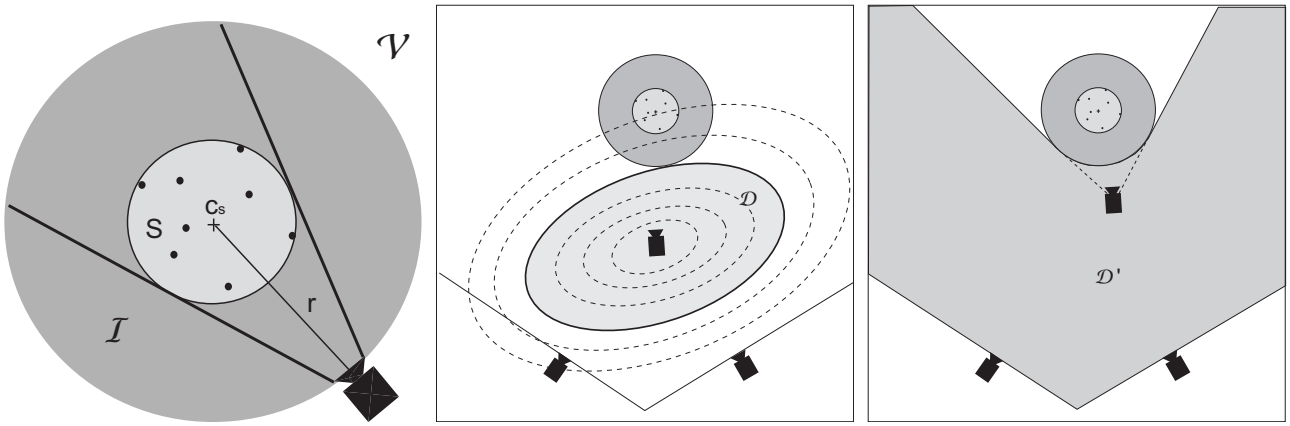
Note that the proposed control law is independent of the current actual camera position since J_d and Γ_d (and so Γ_d^{-1}) are constant matrices for a given target. We require that the desired camera can be triangulated, namely $c_d \in \mathcal{W}$ where $\mathcal{W}$ is defined in (6). We triangulate the position of the target position c_d with respect to the reference cameras, just one time at the beginning of the process, in order to obtain the constant matrix Γ_d and then the control input (20). Note that since (20) uses only the desired Jacobian, we need not compute $c_a \in \mathcal{W}$.

B. "Global" convergence of the translational controller

Following [7], we demonstrate the stability using Lyapunov analysis, and we compute a conservative domain of attraction of our control law.

Suppose that $c_d \in \mathcal{W}$ and the actual camera translation can be fully actuated, namely

$$\dot{c}_a = u. \quad \Longrightarrow \quad \dot{\tilde{c}} = \dot{c}_d - \dot{c}_a = -u. \tag{21}$$



(a) (b) (c)

Fig. 4. (a) The field of view defines the invisible set $\mathcal{I}$. Inside the invisible set $\mathcal{I}$ the camera cannot maintain the whole bounding sphere S within the FOV. The visible set, $\mathcal{V} = \mathbb{R}^3 - \mathcal{I}$, guarantees that the camera can keep all the features in the FOV. (b) Lyapunov surfaces and the "safe" domain of attraction $\mathcal{D}$. (c) Less conservative, but approximate domain of attraction, $\mathcal{D}'$.

A suitable Lyapunov function may be chosen as

$$V(c_a) := \frac{1}{2}\tilde{c}^T J_d^T \Gamma^2 J_d \tilde{c}. \tag{22}$$

Note that if $c_d \in \mathcal{W}$, then V is positive definite on $\mathcal{L}$, since $\tilde{c}$ is equal to zero only when $c_a = c_d$. Since J_d and Γ_a are constant matrices, differentiating (22) yields

$$\dot{V} = \tilde{c}^T J_d^T \Gamma_a^2 J_d \dot{\tilde{c}} = -\tilde{c}^T J_d^T \Gamma_a^2 J_d u.$$

Imposing (20) as control input and recalling the pseudo inverse defined in (19)

$$\begin{aligned}\dot{V} &= -\tilde{c}^T J_d^T \Gamma_a^2 J_d J_d^{\dagger} \Gamma_d^{-1} \tilde{e} \\ &= -\tilde{c}^T (J_d^T \Gamma_a^2 J_d)(J_d^T \Gamma_d^2 J_d)^{-1} J_d^T \Gamma_d^2 \Gamma_d^{-1} \tilde{e} \\ &= -\tilde{c}^T J_d^T \Gamma_d \tilde{e}.\end{aligned}$$

Since the diagonal matrices Γ and Γ_d commute, substitute $\tilde{e}$ from (17) to obtain

$$\dot{V} = -\tilde{e}^T \Gamma_a \tilde{e}$$

which is negative definite in c_a on the set $\mathcal{L}$, so long as $c_d \in \mathcal{L}$. Since the ellipsoidal level sets of V are positive invariant with respect to time, a conservative estimate for the domain of attraction is given by the interior of the largest level set completely within $\mathcal{L}$.

C. Rotational controller

During VS, some features may leave the FOV of the actual camera, thus crippling our ability to estimate the epipolar geometry. Since our translational controller relies on the visibility of the features in the scene to recover the epipoles, we control the camera orientation to maintain the scene points within the FOV.[1] After the translational movement is completed we then align the camera to the same target orientation.

Assume that all scene points lie within the FOV of the desired view, $\{d\}$, the two reference views, $\{1, 2\}$, as well as the initial configuration actual camera, $\{a\}$. Define the smallest bounding circle containing all the features in the actual image plane, and denote its center by $\overline{p} = [\overline{p}_{(1)}, \overline{p}_{(2)}]^T$. We control the camera orientation about the X and Y axes so as to maintain the bounding circle in the FOV, by keeping it centered on the image plane. In addition, we consider the angle, θ, generated by two points on the target. For numerical conditioning, it is advantageous to select the longest line segment that can be constructed from features points and allowing that this may change during the motion [12]. Thus, we propose a local image-based coordinate system [13] that relates the three camera orientation DOFs and the three image DOFs given by centroid position and the feature angle. This is similar to the "hybrid" feature based control by Corke and Hutchinson [12]. We consider $0 \leq \theta_{ij} < 2\pi$ as the angle between the u axis of the image plane and the line segment joining features points i and j. The angular velocity can therefore be commanded by the heuristic controller

$$\begin{aligned}\omega_x &= \lambda_1(\overline{p}_{(2)} - v_0) \\ \omega_y &= -\lambda_2(\overline{p}_{(1)} - u_0) \\ \omega_z &= \lambda_3(\theta^*_{ij} - \theta_{ij}).\end{aligned} \tag{23}$$

where λ_1, λ_2, λ_3 are positive scalar gains, and (u_0, v_0) is the image center.

Once the camera arrives at the correct position (via the translational controller), we switch to another rotational controller in order to obtain the same alignment between the two views. Let $\overline{p}_d = [\overline{p}_{d(1)}, \overline{p}_{d(2)}]^T$ denote the centroid of the desired features. For the first two component we simply adopt a variation of the controller presented in (23) where instead of p_0 we consider $\overline{p}_d$. The rotational controller becomes thus

$$\begin{aligned}\omega_x &= \lambda_1(\overline{p}_{(2)} - \overline{p}_{d(2)}) \\ \omega_y &= -\lambda_2(\overline{p}_{(1)} - \overline{p}_{d(1)}) \\ \omega_z &= \lambda_3(\theta^*_{ij} - \theta_{ij}).\end{aligned} \tag{24}$$

where θ_{ij} represents the angle computed over the desired features and λ_3 a suitable scalar gain.

[1] Note that while we do require that the camera centers remain in $\mathcal{L}$, they may freely depart the FOV without consequence, since their projection as epipoles are "virtual" features computed from the epipolar geometry.

The translational and rotational controllers are completely decoupled. However, the location of the features on the image is affected by the translational controller, an effect that our heuristic controller (23) treats as a disturbance. In practice we "turn up" the rotational gains to ensure that the rotational controller successfully maintains the features within the FOV at all times, so that the epipolar geometry can be estimated for the translational controller. The controller in (24) does not suffer from the same problem since when it starts to work, the translation movement has already been performed and the actual camera is stationary on the target location.

Currently our rotational control strategy is ad hoc. Formalizing the strategy, by more carefully analyzing the basins of attraction of the constituent rotational controllers, and more carefully stiching them together [14], represents work in progress.

D. "Safe" domain of attraction that respects the FOV

As described in Sec. III-B, the control input (20) guarantees convergence for all initial conditions of c_a within the largest level set of V (22) that is completely in front of both cameras, so long as the goal position is triangulable, namely $c_d \in \mathcal{W}$. However all such initial conditions do not guarantee all features will lie within the FOV.

Consider a set of points in space and define the bounding sphere S, with center c_s, as the smallest sphere that contains all the points. Let r denote the minimum distance from c_s such that the camera is able to maintain S in the field of view. We define the invisible set $\mathcal{I} \subset \mathbb{R}^3$ as an open ball centered around c_s of radius r. For every location outside $\mathcal{I}$, there exists a camera orientation that is able to see the whole bounding sphere S. The visible set $\mathcal{V} \subset \mathbb{R}^3$ (see Fig. 4.a) is given by the complement of $\mathcal{I}$, namely

$$\mathcal{V} = \mathbb{R}^3 - \mathcal{I}. \tag{25}$$

For a given starting point $p \in \mathcal{L} \cap \mathcal{V}$, and goal $c_d \in \mathcal{L} \cap \mathcal{W}$, the actual camera trajectory may intersect $\mathcal{I}$ causing the transient lost of features from the FOV. Fortunately, the Lyapunov function V (22) specifies a set of positive time invariant sets

$$\mathcal{D}_k = \{c_a \in \mathcal{L} : V(c_a) \leq k\}.$$

The "safe domain", $\mathcal{D}$, is the largest such set wholly contained in $\mathcal{L} \cap \mathcal{V}$. Clearly, any initial condition inside $\mathcal{D}$ guarantees that the camera trajectory will converge to the desired location, while staying in the visible set. Fig. 4.b, shows the set of level surfaces with the safe domain of attraction $\mathcal{D}$.

We have found $\mathcal{D}$ to be quite conservative. Camera trajectories nearly follow straight lines (refer to Fig. 5.c), and thus we define

$$\mathcal{D}' = \{c_a \in \mathbb{R}^3 : \overline{c_a c_d} \subset \mathcal{L}\}$$

where $\overline{c_a c_d}$ represents the segment between c_a and c_d (refer to Fig. 4.c). We suspect that the true domain of attraction is very similar to $\mathcal{D}'$.

IV. EXPERIMENTS

The multi-view system consists of four images: a pair of reference images, a desired image and the actual camera image at each time step. The epipolar geometry estimated from the corresponding points in the images allow us to control the actual camera to the desired location with a large domain of attraction. We executed a sequence of simulations starting with perfect knowledge of the baseline between the reference views, and noiseless image data, and subsequently adding error to our knowledge of the reference baseline and image measurement noise. We assumed no prior knowledge about the 3D scene, however we assume the internal camera parameters are calibrated.

The algorithm proceeds as follows. At the beginning of the process, store the actual image as the first reference image. Then, let the camera undergo a random translation such that the features remain in FOV. Then, store the second reference image. Generate a "good" initial condition (within $\mathcal{D}'$). To initialize the system, compute the epipoles, e_d, between the reference frames and the desired frame. Also, using the reference baseline, triangulate $c_d = g^{\dagger}(e_d)$, which is required to compute the (constant) matrix Γ_d. In real-time, execute the translational (20) and rotational (23) control laws, using at each instant the latest estimate of e_a. Once the target is gained switch on the controller (24) in order to obtain the same target orientation. Fig. 5.a and 5.b show the actual camera translational and orientation error. Camera and feature trajectories can also be observed in Fig. 5.c and Fig. 5.d.

We have found that the method exhibits very little sensitivity to errors in the baseline between the reference images. We tested this for errors up to 50% of the baseline length, with only modest degradation of the Cartesian trajectory (and no degredation of the asymptotic performance), when compared to error-free experiments. As also noted in [15], stereo systems are typically more sensitive to orientation miscalibration. Since our translational controller is based on that of Hager et al. [7], and they report a comprehensive set of experiments with erroneous orientations between the two views, we have concentrated our attention on the sensitivity of the algorithm with respect to image noise.

We added noise to the image feature points which deteriorates the epipole estimates; it is well known that essential matrix estimation is quite sensitive to noisy data [16], [17].

We modeled our camera with a resolution of 640×480. We selected initial locations of c_a at random from $\mathcal{D}'$. We added an uniformly distributed error to each point as a percentage of the variance, σ, of the cloud of features in the image.

Table I shows the percentage of convergent experiments versus the noise added to the feature points for 700 experiments. The first column represents the noise added, expressed as percentage of variance of the features on the image. The second column shows the typical range of resulting absolute pixel error added to each point in

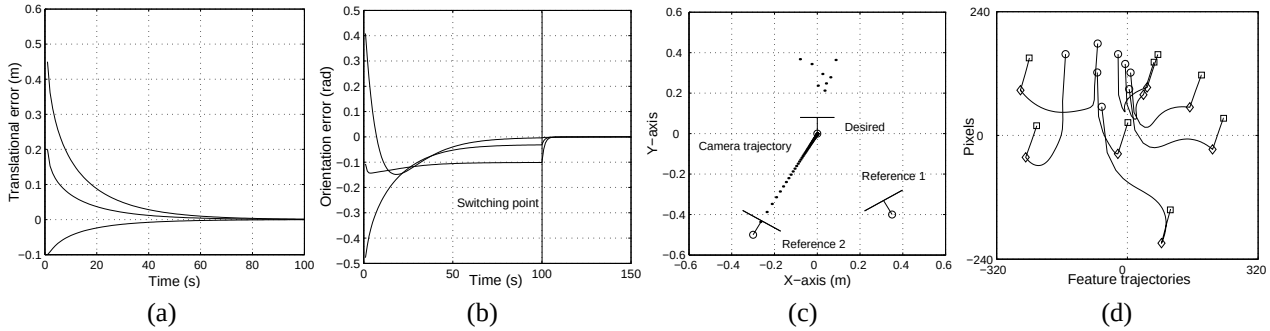


Fig. 5. Simulated experiments. (a) Camera translational error with free-noise data. (b) Actual camera orientation error. Note at the instant 100 the rotational control switches to align the actual camera orientation to the target camera orientation. (c) Actual camera trajectory. (d) Feature trajectories: actual features at the beginning of the process (○); desired features (□); actual features at the switching instant (◇).

added noise	typical pixel noise range	convergence
0%	0	100%
5%	± 1.2	100%
10%	± 2.4	95%
15%	± 3.6	77%
20%	± 4.8	53%
25%	± 6.1	22%
30%	± 7.2	12%

TABLE I

PERCENTAGE OF CONVERGENCE WITH NOISY DATA

our experiments. The last column shows the percentage of trials that converged after a fixed simulation time.

V. CONCLUSION

Rather than relying on only two views of a scene, the "actual" and "desired" views, this paper suggests a simple method to incorporate additional prior views to generate a globally convergent VS system that maintains feature visibility. In the absence of noise, any two reference configurations with a known, nonzero baseline disparity guarantee the convergence of our algorithm. Our simulations suggest that large uncertainty in the baseline and noisy image data can be tolerated.

Ultimately, a VS system could learn as it progresses, taking into account information from *all* prior views of a scene. For example, geometric model reconstruction using multi-view structure-from-motion (SFM) algorithms [18] could provide a robust and statistically sound means by which to recover specific information, such as feature depth. Moreover, we suspect that judiciously controlled motion, especially during the learning phase, could greatly enhance a system's robustness with respect to measurement noise and uncertainty in the camera or motion models.

REFERENCES

[1] E. C. Sobel, "The locust's use of parallax to measure distance," *J. Comp. Physiol.*, vol. 167, pp. 579–588, 1990.

[2] G. Hager, D. Kim, A. Rizzi, and D. Koditschek, "A "robust" convergent visual servoing system," in *In Proceedings of the International Conference on Intelligent Robots and Systems*, vol. 1, 1995, pp. 348–353.

[3] P. Rives, "Visual servoing based on epipolar geometry," in *International Conference on Intelligent Robots and Systems*, vol. 1, 2000, pp. 602–607.

[4] R. Basri, E. Rivlin, and I. Shimshoni, "Visual homing: Surfing on the epipoles." in *ICCV*, 1998, pp. 863–869.

[5] E. Malis, F. Chaumette, and S. Boudet, "2d 1/2 visual servoing," *IEEE Transaction on Robotics and Automation*, vol. 15, pp. 328–250, Apr. 1999.

[6] C. J. Taylor, J. P. Ostrowski, and S. H. Jung, "Robust visual servoing based on relative orientation," in *IEEE Conf. on Comp. Vision and Patt. Recog*, June 1999, pp. 574–580.

[7] D. Kim, A. A. Rizzi, G. H. Hager, and D. E. Koditschek, "A "robust" convergent visual servoing system," *In Proc. International Conference on Intelligent Robots and Systems*, vol. I, pp. 348–353, 1995.

[8] A. A. Rizzi and D. E. Koditscheck, "An active visual estimator for dexterous manipulator," *IEEE Transaction on Robotics and Automation*, vol. 12, no. 5, 1996.

[9] O. D. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, 1993.

[10] R. Hartley and A. Zisserman, *Multiple view in computer vision.* Cambridge University Press, September 2000.

[11] R. Hartley, "In defence of the 8-point algorithm." in *Proc. of IEEE Int. Conference on Computer Vision*, Cambridge MA,USA, June 1995, pp. 1064–1070.

[12] P. Corke and S. Hutchinson, "A new partitioned approach to image-based visual servo control," in *IEEE Transaction on Robotics and Automation*, vol. 17(4), 2001, p. 507515.

[13] N. J. Cowan and D. E. Chang, "Geometric visual servoing," *IEEE Transactions on Robotics*, 2004, in review.

[14] R. R. Burridge, A. A. Rizzi, and D. E. Koditscheck, "Sequential composition of dynamically dexterous robot behaviors." *Int. J. Rob. Res.*, vol. 18(6), pp. 534–555, 1999.

[15] G. D. Hager, W. Chang, and A. S. Morse, "Robot hand-eye coordination based on stereo vision," *IEEE Control Systems*, vol. 15, no. 1, pp. 30–39, Feb. 1995.

[16] Q.-T. Luong, R. Deriche, O. Faugeras, and T. Papadopoulo, "On determining the fundamental matrix: analysis of different methods and experimental results." in *Artificial Intelligence Journal*, vol. 78, October 1995, pp. 87–119.

[17] Q. Luong and O. Faugeras, "The fundamental matrix: theory, algorithms and stability analysis." *Int. Journal of Computer Vision*, vol. 17, no. 1, pp. 43–76, 1996.

[18] S. J. Maybank, *Theory of reconstruction from image motion*. Berlin: Springer Verlag, 1993.