(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 July 2010 (08.07.2010)

PCT

(10) International Publication Number
WO 2010/077389 A2

(51) International Patent Classification:
H04N 7/18 (2006.01) *GOSB 11/00* (2006.01)
G06K 9/00 (2006.01)

(21) International Application Number:
PCT/US2009/052803

(22) International Filing Date:
5 August 2009 (05.08.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/086,290 5 August 2008 (05.08.2008) US

(71) Applicant (*for all designated States except US*): UNIVERSITY OF FLORIDA RESEARCH FOUNDATION, INC. [US/US]; 223 Grinter Hall, Gainesville, FL 3261 1 (US).

(72) Inventors; and
(75) Inventors/Applicants (*for US only*): GANS, Nicholas, R. [US/US]; 4653 SW 44th Lane, Gainesville, FL 32698 (US). DIXON, Warren [US/US]; 8681 SW 89th Lane, Gainesville, FL 32698 (US).

(74) Agent: RISLEY, David; Thomas, Kayden, Horstemeyer & Risley, LLP, 600 Galleria Parkway, Suite 1500, Atlanta, GA 30339 (US).

(81) Designated States (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (*unless otherwise indicated, for every kind of regional protection available*): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report (Rule 48.2(g))*

WO 2010/077389 A2

(54) **Title:** SYSTEMS AND METHODS FOR MAINTAINING MULTIPLE OBJECTS WITHIN A CAMERA FIELD-OF-VIEW

(57) **Abstract:** In one embodiment, a system and method for maintaining objects within a camera field of view include identifying constraints to be enforced, each constraint relating to an attribute of the viewed objects, identifying a priority rank for the constraints such that more important constraints have a higher priority that less important constraints, and determining the set of solutions that satisfy the constraints relative to the order of their priority rank such that solutions that satisfy lower ranking constraints are only considered viable if they also satisfy any higher ranking constraints, each solution providing an indication as to how to control the camera to maintain the objects within the camera field of view.

SYSTEMS AND METHODS FOR MAINTAINING MULTIPLE OBJECTS WITHIN A CAMERA FIELD-OF-VIEW

NOTICE OF GOVERNMENT-SPONSORED RESEARCH

The disclosed inventions were made with U.S. Government support under Grant No.: DE-FG04-86NE37967 awarded by the U.S. Department of Energy. The U.S. Government has rights in the claimed inventions.

CROSS-REFERENCE TO RELATED APPLICATION

This application claims priority to co-pending U.S. provisional application entitled "Visual Servo Controller Configured to Keep Multiple Objects in a Camera Field-of-View" and having serial number 61/086,290, filed August 5, 2008, which is entirely incorporated herein by reference.

BACKGROUND

The use of image data in feedback control of a mechanical system, commonly referred to a visual servo control, is an established and diverse field. There are many approaches, including image-based visual servoing (IBVS), position-based visual servoing, partitioned methods, and switching methods. Most visual servo control methods have a common goal, such as a goal image, that is used to regulate a camera, robot manipulator, or unmanned vehicle to a desired camera pose or trajectory. For example, an image of a scene may be captured (the goal image) and later images may be captured and compared with the original image after the camera has been moved to identify when the camera has returned to its original position (the goal pose).

In many vision-based control tasks, there is no goal image from which to work, as well as no goal pose or trajectory. For instance, if the desired task is to maintain multiple moving objects in the camera filed-of-view, no goal image exits because the motion and position of the objects are not known ahead of time. In such situations, visual servo control methods may be ineffective.

BRIEF DESCRIPTION OF THE DRAWINGS

The accompanying drawings, which are incorporated in and form a part of the specification, illustrate embodiments of the presently disclosed inventions. The drawings are not necessarily drawn to scale.

FIG. 1 is a diagram that illustrates an example camera model.

FIG. 2 is a flow diagram of an example method for maintaining multiple moving objects within a camera field of view.

FIG. 3 is a graph that depicts feature point trajectories for tracking randomly moving points with PID control.

FIG. 4 comprises graphs that depict error in Φ_w, Φ_v, Φ_P over time for tracking randomly moving points with PID control.

FIG. 5 comprises graphs that depict camera velocity over time for tracking randomly moving points with PID control.

FIG. 6 is a graph that depicts feature point trajectories for tracking two objects with PID control.

FIG. 7 comprises graphs that depict error in Φ_m, Φ_v, Φ_p over time for tracking two objects with PID control.

FIG. 8 comprises graphs that depict camera velocity over time for tracking two objects with PID control.

FIG. 9 is a schematic diagram of an example system that can be used to maintain multiple objects within a camera field of view.

FIG. 10 is a block diagram that illustrates an example architecture for a computer system shown in FIG. 9.

DETAILED DESCRIPTION

Introduction

As described above, existing visual servo control methods may be ineffective at performing various vision-based control tasks, including maintaining multiple moving objects within a camera field of view (FOV). Described herein are systems and methods with which such objects can be effectively maintained within the camera FOV. Unlike conventional systems and methods, the disclosed systems and methods use no goal images. Instead, a set of underdetermined task functions are used to regulate various imposed constraints that control one or more of the camera's, settings, pose, position, and velocity so as to maintain the target objects within the camera FOV. The imposed constraints can be selected to suit the particular application. Example constraints include the average position or mean of the objects and the variance of each object from the mean. Of course, other constraints can be used depending upon the application or the degree of control that is to be exercised.

In some embodiments, a desired task function velocity is mapped to a time-varying feature velocity through corresponding underdetermined task function Jacobians featuring multidimensional null spaces. A multidimensional null space permits cascading of several task functions, which generates a feature velocity. These task function Jacobians are suitable for task-priority kinematic control. In some cases, the time-varying feature velocity is mapped to camera motions using IBVS methods.

The above-described control approach can be used for various purposes. One possible scenario is surveillance of a crowd in which a camera views the crowd from above and tracks individuals within the camera's FOV. Another scenario involves tracking moving vehicles.

In the discussions that follow, a control approach is first described that uses various task functions that can be used to generate the desired feature velocity. An application of the approach and simulations of tasks are also described below to demonstrate the performance of the control approach.

Discussion of the Control Approach

1. *Camera Model*

Consider a camera with coordinate frame $\mathcal{F}_c(t)$ as shown in FIG. 1. The camera views a collection of k feature points in front of the camera. These points have coordinates M_1(V) e R^3 defined as

$$M_i = [X_i, Y_i, Z_i]^T, \ \forall i \in \{1 \ldots k\}$$

in the camera frame. An image of the points is captured, resulting in a projection to a set of points in the image frame. These image points are given by the normalized coordinates

$$\tilde{m}_i = [\frac{X_i}{Z_i}, \frac{Y_i}{Z_i}, 1]^T = [\text{w,,l}] \quad {}^{\text{r}}, \text{ Vi} \in \{\text{l...*}\}.$$

Because the last element of the three-dimensional normalized coordinates is superfluous, it is not considered in the sequel. Define the coordinates $m, (t) \in E^2$ as

$$m_1 = [W,]^T \quad (1)$$

with velocity $\dot{m}, (t)$ e R^2 in the image plane given by

$$\dot{m}, = [*., \dot{y}_i]^T . \quad (2)$$

Given the collection of *k* feature points, along with their coordinates and velocity vectors, a state position vector *m(t)* and state velocity vector *m(i)* are defined as

$$m = [m_1^T, m_2^T, \cdots, m_k^T]^T$$
$$\dot{m} = [\dot{m}_1^T, \dot{m}_2^T, \cdots, \dot{m}_k^T]^T .$$

The features points velocity is given as a function of the camera velocity $v_c(t) = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z, f$ e l ^{6} by the relationship

$$\dot{m} = Lv_c, \quad (3)$$

where Z(f) $\in \mathbb{R}^{2Ax6}$ is the image Jacobian, $v_x(t)$, $v_y(t)$, and $v_z\{t)$ describe the linear velocity of the camera, and $\omega_x(i)$, $\omega At)$, and $\omega_z(t)$ *are* the angular velocity of the camera for a six degree of freedom system. The image Jacobian for the feature points, $L(t) \in \mathbb{R}^{2k\times 6}$, is given by concatenating a set of k submatrices $L_i(t) \in \mathbb{R}^{2\times 6}$, with $L_i(t)$ given as

$$L_i = \begin{bmatrix} \frac{1}{z_i} & 0 & \frac{x_i}{z_i} & -x_i y_i & 1+x_i^2 & -y_i \\ 0 & \frac{1}{z_i} & \frac{y_i}{z_i} & -1-y_i^2 & x_i y_i & x_i \end{bmatrix}. \tag{4}$$

In the case that the feature points are not static in an inertial frame world frame (e.g., the feature points are tracked on moving objects), the time derivative of the feature points is given by

$$\dot{m} = Lv_c + \varepsilon, \tag{5}$$

where $\varepsilon(t)$ is generally an unknown function. It is assumed that $\varepsilon(t)$ is bounded.

2. *Task Function-Based Kinematic Control*

The control objective is to keep a set of feature points within the camera FOV without a specific camera pose. To achieve this, a set of task functions are defined using image feature coordinates. By regulating the task functions, features can be kept in the FOV. However, the task functions may compete in the sense that reducing the error of one task function may increase the error of another. To avoid competition between task functions, task-priority kinematic control is used. This section provides a foundation for the task functions, how they are combined in task-priority kinematic control, and how the resulting signal is mapped to camera velocity commands. Three specific task functions and final control law are detailed in the section entitled Control Development.

Let $<p(t) e R"$ denote a task function of the feature point coordinates $m_i(t)$ as

$$\phi = f(m_1,...,m_k)$$

with derivative

$$\dot{\varphi} = \sum_{i=1}^{k} \frac{d f}{dm,} \dot{m}_i = J(\dot{m})\dot{m}, \tag{6}$$

where $J\{m) \in W^{*2k}$ is the task Jacobian matrix. In the section entitled Control Development that follows, the task functions are of dimension $« \leq 2$.

The task is to drive the feature points along a desired velocity $\dot{m}_\phi(f)$ such that $\phi(t)$ follows a desired trajectory $\phi_d(t)$. Given the underdetermined structure of the Jacobian matrix, there are infinite solutions to this problem. The typical solution based on the minimum-norm approach is given as

$$\begin{aligned} \dot{m}_\varphi &= J^\dagger\left[\dot{\varphi}_d - \lambda(\varphi - \varphi_d)\right] \\ &= J^\tau (jJ^\tau)^{-1} [\phi_d - \lambda(cp - \phi_d)l \end{aligned} \tag{7}$$

where λ is a positive scalar gain constant and $J^\dagger(m) e R^{2kx"}$ denotes the minimum-norm general inverse of $j(m)$. The camera velocity $v_c(t)$ is designed as

$$v_c = L^+\dot{m}_\phi = (L^\tau LYL^\tau \dot{m}_\phi, \quad \textbf{(8)}$$

where $L^+(/) \in R^{6*2^k}$ denotes the least squares general inverse used for *L{t).* Note that since *j(m)* is underdetermined (i.e. more columns than rows), and *Lit)* is overdetermined (i.e. more rows than columns), the general inverses have different solutions and care is taken to denote them differently. The symbol f denotes the minimum norm general inverse used for *j{t)* and + denotes the least-squares general inverse used for *L(*ή*.*

A set of task functions can be combined in various ways. The simplest approach is to add the desired feature velocity from each task function to create the total desired feature point velocity. For example, consider two tasks $\phi_a(t)$ and $\phi_b\{t)$ with resulting desired feature velocities $m_a\{t)$ and $m_h(t)$. The camera velocity is then given as

$$v_c = Ü \ (\dot{m}_a + \dot{m}_{\phi b}). \quad \textbf{(9)}$$

Since the task function velocities are undetermined, an optional method is to choose one task as primary, and project the other tasks into the null space of the primary task derivative as

$$v_c = L^+\left(\dot{m}_a + (I - J_a^\dagger J_a)\dot{m}_b\right)$$

$$= \mathrm{r} \ \ (J_a^\dagger \dot{\varphi}_\alpha + (/ - JIJj \ J_b V_b), \quad \textbf{(10)}$$

where $J_a(m_a)$ and $J_b(m_b)$ are the task Jacobian matrices with respect to $\phi_a(t)$ and $\phi_b\{t)$, respectively.

The approach in Equation 10 prevents the systems from competing and negating each other, as the primary task is always accomplished. Lower priority control tasks are achieved to the extent that they do not interfere with higher priority tasks. Tertiary, quaternary, etc. tasks can be prioritized by repeating this process and projecting each subsequent task into the null space of the preceding task Jacobians.

3. *Control Development*

In this section, three task functions are presented as part of a task-priority kinematic controller. The control objective of this invention is to keep a set of feature points corresponding to multiple tracked, moving targets within the camera FOV. Two task functions are designed to regulate the mean and variance of feature point coordinates. Regulating the mean at the camera center keeps the feature points centered in the FOV. Regulating the variance restricts the distance between the feature points and keeps features away from the edge of the FOV. The third task function maximizes motion perceptibility, which ensures desired image velocities can be met. These three task functions are cascaded through null space projection and mapped to camera velocity, as described the section entitled Task Function-Based Kinematic Control.

a. *Task Function for Mean of Image Points*

Controlling the mean of feature point coordinates helps to ensure the feature points are centered around a position in the image plane. Let $\phi_m(t) \in \mathbb{R}^2$ denote a task function defined as the sample mean

$$\phi_m = \frac{1}{k}\sum_{i=1}^{k} m_i = \overline{m}. \tag{11}$$

The open-loop derivative of $\phi_m(t)$ is given by

$$\dot{\phi}_m = \frac{1}{k}\sum_{i=1}^{k} \frac{\partial \phi_m}{\partial m_i}\dot{m}_i = J_m \dot{m} \tag{12}$$

$$= J_m(Lv_c + \varepsilon), \tag{13}$$

where $J_m(t) \in R^{2x2k}$ is a task function Jacobian, and L, v_c, and ε were given in Equation 5. The Jacobian $J_m(t)$ is defined as

$$J_m = \frac{1}{k}[I_2, \cdots, I_2],$$

where I_2 is the $2\chi 2$ identity matrix and is repeated k times.

In the general case, the objective is to regulate the mean to a desired trajectory. This helps ensure the features are centered around a specific, time varying point in the image. A desired task function trajectory $\phi_{md}(t)$ is defined with a known, smooth derivative $\dot{\phi}_{md}(t)$. To regulate the mean to the desired trajectory, a feature point velocity, denoted by $\dot{m}_m(t) \in R^{2k}$, can be designed as

$$\dot{m}_m = J_m^\dagger \left[\dot{\varphi}_{md} - \lambda_m(\varphi_m - \varphi_{md})\right], \qquad (14)$$

where λ_m is a positive scalar gain constant, and the subscript denotes that $\dot{m}_m(t)$ is designed with respect to the task function for mean. Combining Equations 12 and 14 gives the closed-loop derivative of $\phi_m(t)$ as

$$\dot{\Phi}_m = -\lambda_m(\Phi_m - \Phi_{md}) + \dot{\Phi}_{md} + \varepsilon_m, \qquad (15)$$

where $\varepsilon_m(t) = J_l(t)\varepsilon(t) \in \mathbb{R}^2$ is an unknown function of the velocity of the feature points, and can be modeled as an input disturbance. Assuming that $\varepsilon_m(t)$ is bounded, and λ_m is significantly large, it can be shown that $\phi_m(t)$ goes to a neighborhood of $\phi_{md}(t)$ exponentially fast. If the desired trajectory $\phi_{md}(t)$ is simplified to a constant set value ϕ_{md}, Equations 14 and 15 can be simplified as

$$\dot{m}_m = -\lambda_m J_m^\dagger(\phi_m - \phi_{md}) \qquad (16)$$

and

$$\dot{\phi}_m = -\lambda_m(\phi_m - \phi_{md}) + \varepsilon_m.$$

The controller can be made more robust to the disturbance ε_m by adding an error integral term and an error derivative term to Equation 16. This proportional-integral-derivative (PID) controller is given by

$$\dot{m}_{m} = -J_l^{+}\left(\lambda_j \phi_{me} + \lambda_{m} \int \phi_{me}\, dt + \lambda_{md} \frac{d}{dt}\phi_{me}\right), \tag{17}$$

where $\Phi_{me}(t) = \Phi_m(t) - \phi_{md}$ and $\lambda_m, \lambda_{md} \in R$ are constant gains.

For more than three feature points, the desired $\dot{m}_m(t)$ may not be achievable. Substituting $\dot{m}_m(t)$ in Equation 3 gives

$$v_c = V\dot{m}_m.$$

If $k > 3$, $L^{+}(t)$ is underdetermined, meaning there are six degrees of freedom available in $v_c(t)$ to control more than six terms in $\dot{m}_m(t)$. This problem is partially addressed by maximizing perceptibility in the section entitled Task Function for Perceptibility of Image Points.

b. *Task Function for Variance of Image Points*

Regulating the variance of the feature point coordinates regulates the spread of the feature points in the image. That is, regulating the variance controls how far the feature points drift from the mean value. Define a task function $\phi_v(t) \in R^2$ as the sample variance

$$\phi_v = \frac{1}{k}\sum_{i=1}^{k}\begin{bmatrix}(x_i-\bar{x})^2\\(y_i-\bar{y})^2\end{bmatrix},$$

where *x(t)* and *y(t)* are the mean of all the *x* and *y* components of /«,(r),z e{l...&} , respectively.

To find the variance task function Jacobian, consider the partial derivative of ϕ*_V*(ή with respect to *x*$_\lambda$*(t)*

$$\frac{\partial\psi\pounds}{dx_{,\lambda}} = \frac{\partial}{Sx_1}\left[\frac{1}{k}\sum_{i=1}^{k}(x_i-\bar{x})^2\right]$$

$$= \frac{2}{k}\left[\sum_{i=1}^{k}(x_i-\bar{x})\left(\frac{\partial x_i}{\partial x_1}-\frac{1}{k}\sum_{j=1}^{k}\frac{\partial x_j}{\partial x_1}\right)\right]$$

$$= \frac{2}{k}\left[(x_1-\bar{x})-\frac{1}{k}\sum_{i=1}^{k}(x_i-\bar{x})\right]$$

$$=: \frac{?}{\kappa}[(X_j - \bar{x}) - \bar{x} + x]$$

$$= \frac{2}{k}[(x_1 - \bar{x})]- \tag{18}$$

Repeating the above simplifications for all *x*$_t$*(i), yX*ή, *i* ∈ *{l...k}* , gives the open-loop derivative of ϕ*_V(t)* as

$$\dot{\varphi}_v = J_v\dot{m} \tag{19}$$

$$= J_V(Lv_C + \varepsilon) \tag{20}$$

where and $L(t)$, $v_c(t)$, and $\varepsilon(t)$ were given in Equation 5, and $J_v(t) \in \mathbb{R}^{2x2k}$ is a task function Jacobian given by

$$J_v = \frac{2}{k}\begin{bmatrix} x_1 - \bar{x} & 0 & x_2 - \bar{x} & 0 \\ 0 & y_1 - \bar{y}, & 0 & y_2 - \bar{y} \end{bmatrix} , \ldots, \begin{matrix} x_k - \bar{x} & 0 \\ 0 & y_k - \bar{y} \end{bmatrix}.$$

To regulate the variance to a desired trajectory $\phi_{vd}(t)$ (with a known, smooth derivative $\dot{\phi}_{vd}(t)$) the feature point velocity $\dot{m}_v(t) \in \mathbb{R}^{2k}$ can be designed by following the method in the section entitled Task Function for Mean of Image Points to give

$$\dot{m}_v = J_v^{\dagger}\left[\dot{\varphi}_{vd} - \lambda_v(\varphi_v - \varphi_{vd})\right], \tag{21}$$

where λ_v is a positive scalar gain constant. Combining Equations 19 and 21 gives the closed-loop derivative of $\phi_v(t)$ as

$$\dot{\phi}_v = -\lambda_v(\phi_v - \phi_{vd}) + \dot{\phi}_{vd} + \varepsilon_v,$$

where ε,(f)eR ^{2} is an unknown disturbance caused by the velocity of the feature points. Under the assumption that that $\varepsilon_v(t)$ is bounded, if λ_v is significantly large, it can be shown that $\phi_v(t)$ goes to a neighborhood of $\phi_{vd}(t)$ exponentially fast.

If the desired trajectory $\phi_{vd}\{t)$ is simplified to a constant set value ϕ_{vd} > Equations 14 and 15 can be simplified as

$$\dot{m}_v = -\lambda_m J_m^{\dagger}(\Phi_m - \Phi_{md}) \tag{22}$$

and

$$\dot{\phi}_v = -\lambda_v(\phi_v - \phi_{vd}) + \varepsilon_v$$

To increase robustness to the input disturbance $\varepsilon_V(t)$, integral and derivative terms can be added to Equation 16 to give the PID feedback control

$$\dot{m}_v = -J_v^{\dagger}\left(\lambda_v \varphi_{ve} + \lambda_{vi}\int_0^t \varphi_{ve}\,dt + \lambda_{vd}\frac{d}{dt}\varphi_{ve}\right), \tag{23}$$

where $\Phi_{ve}\{t) = \phi_V\{t) - \phi_{vd}$ and $\lambda_{vi}, \lambda_{vd} \in R^+$ are constant gains.

c. *Task Function for Perceptibility of Image Points*

Perceptibility gives a measure of how well a camera can perceive the motion of objects in the FOV. Roughly speaking, if perceptibility is high, small object or camera velocities results in notable feature velocities in the image plane (e.g., high optical flow). This is especially important if there are more than three feature points, as the available feature point velocities are constrained due to an overdetermined image Jacobian. Maintaining a high perceptibility helps to ensure a larger span of available feature point velocity vectors.

Perceptibility is a scalar function of the image Jacobian *Lit),* defined as

$$w_v = \sqrt{\det(L^T L)} = \prod_{i=1}^{6} \sigma_i ,$$

where σ,(*t*Je I $^{+}$ are the singular values of *L(t).* Maximizing *W_V(t)* is accomplished by maximizing each σ,(r). The matrix L^T (f)i(f)e I ^{6x6} is positive definite and symmetric, so the eigenvalues of *IHf)Uf)* are equal to σ*f(t).* The trace of a matrix is equal to the sum of its eigenvalues. Therefore, the trace of *L^T(t)L{t)* is related to the singular values by

$$Tr(L^{\tau}L) = \sum_{1=1}^{6} \sigma_i^2 .$$

Increasing the trace of *Il{t)L{t)* also increases the perceptibility.

The trace of *Il{t)L{t)* is given by

$$Trill L) = \sum_{i=1}^{k} \frac{2}{Z_i^2} + \frac{x_i^2}{Z_i^2} + \frac{y_i^2}{Z_i^2} + x_i^2 y_i^2 + \{y_i^2 + \backslash)^2$$

$$+ x_i^2 y_i^2 + \left(x_i^2 + 1\right)^2 + x_i^2 + y_i^2$$

$$= \sum_{i=1}^{k} \frac{2}{Z_i^2} + \left(1 + \frac{1}{Z_i^2}\right)\left(x_i^2 + y_i^2\right) + 2x_i^2 y_i^2$$

$$+ (y_i^2 + 1)^2 + (x,^2 + 1)^2 . \qquad (24)$$

For purposes of this disclosure, the task functions are functions of image feature coordinates $m_i(t)$ only. For this reason, the task function for perceptibility contains the terms of $Tr(IIL)$ that do not depend on depth $Z,(t)$. This is acceptable as the depth terms in Equation 24 are all quadratics. Increasing the terms other than $Z,(t)$ serves to increase the trace of $L^T(_t)L(_t)$- A task function for perceptibility of the system is defined as

$$\varphi_p = \frac{1}{\sum_{i=1}^{k} x_i^2 + y_i^2} \tag{25}$$

Since it is desired to increase $Tr(L^1L)$, regulating $\phi_p(t)$ to 0 results in increasing the trace.

The open-loop derivative of $\phi At)$ is given by

$$\dot{\varphi}_p = -4\varphi_p^2 \sum_{i=1}^{6} \begin{bmatrix} \frac{3}{2}x_i + x_i y_i^2 + x_i^3 \\ \frac{3}{2}y_i + y_i x_i^2 + y_i^3 \end{bmatrix}^T \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix}$$

$$= J_p(m)m = J_p(m)\{Lv_c + \varepsilon) \tag{26}$$

where $L(t)$, $v_c(t)$ and $\varepsilon(t)$ were given in (5), and $J_p(m) \in \mathbb{R}^{U2k}$ is the task function Jacobian for perceptibility.

To regulate $\phi_p(t)$ to 0, the feature point velocity $m_p\{t) \in \mathbb{R}^{2k}$ is designed as

$$\dot{m}_p = -\lambda_p J_p^\dagger \phi_p \tag{27}$$

where λ_p is a positive scalar gain constant. Combining Equations 26 and 27 gives the closed-loop derivative of ϕ_p{t) as

$$\dot{\phi}_p = -\lambda_p \phi_p + \varepsilon_p,$$

where $\varepsilon_p(t)$ is an unknown disturbance caused by the velocity of feature points. Despite the presence of the disturbance $\varepsilon_p(t)$, the use of integral feedback is not recommended for the perceptibility term. This is due to the fact that $\phi_p(t)$ is unlikely to ever become zero, leading to possible integrator windup and related stability problems.

d. Cascaded Camera Control Law

As stated in the section entitled Task Function-Based Kinematic Control, the control objective of this invention is to design a kinematic camera controller that maintains a set of feature points within the camera FOV. The mean of feature point coordinates is the best variable to measure the center of the feature points. Regulating the mean is chosen as the primary task in order to keep the feature points centered in the FOV. Regulating the variance to a constant is chosen as the second task to restrict the distance between the feature points and the image center. These two tasks are sufficient to keep features in the FOV. High perceptibility allows these two tasks to work more efficiently by ensuring larger available feature velocities at lower camera velocities. For this reason, increasing perceptibility is chosen as the tertiary task.

The designed feature velocities given in Equations 16, 22, 27, or 17, 23, and 27 are used in the null space projection camera velocity Equation 10 to give the overall controller as

$$v_c = L^{+}\left(\dot{m}_m + \left(I - J_m^{\dagger}J_m\right)\left[\dot{m}_v + \left(I - J_v^{\dagger}J_v\right)\dot{m}_p\right]\right). \tag{28}$$

Application of the Control Approach

As is apparent from the above discussion, the disclosed control approach is a layered approach in which multiple constraints can be enforced according to a predefined priority. An example application of the control approach will now be discussed with reference to the flow diagram of FIGs. 2A and 2B. More particularly, discussed is an example embodiment of a method for maintaining multiple moving objects (targets) within a camera FOV.

Before control can be exercised to maintain objects within the camera FOV, the physical constraints of the system must be identified, as indicated in block 10 of FIG. 2A. Such physical constraints can include whether the camera is a fixed camera or a camera that can physically translate (e.g., pan and tilt) and, if the latter, the range of motion through which the camera can translate. Other physical constraints include the camera specifications, such as focal length, zoom capacity, focusing speed, zoom speed, and the like. In addition, relevant physical constraints can include whether the camera position can be changed, for instance by a helicopter, plane, or unmanned aerial vehicle (UAV) to which the camera is attached and the motion of which such a craft is capable. All such physical constraints affect whether, and the manner in which, objects can be maintained within the camera FOV.

In addition to identifying the physical constraints, the constraints that are to be imposed and enforced must also be identified, as indicated in block 12. In some embodiments, a user can select the imposed constraints to suit to a particular problem with which the user is faced. Alternatively, a default set of imposed constraints can be automatically implemented. As described above, various constraints can be enforced. One constraint that is particularly useful in maintaining multiple moving objects within a camera FOV is the mean constraint, which is used to control the location of the mean object position within the camera FOV. In most cases, control is exercised such that the object mean is coincident with the center of the camera FOV. Another constraint that is useful in the moving objects scenario is the variance constraint, which can be used to control how far each object can stray from the object mean within the camera image. Together, those two constraints can be used to dictate the center of the camera image and a boundary within which the objects are to be contained, which may be thought of as a radius extending from the object mean.

In some cases, the two constraints identified above will be not enough to achieve a desired task, such as ensuring that tracked objects will remain in the camera FOV. Therefore, other constraints can also be used. One such constraint is the perceptibility constraint, which can be used to control the perceptibility of motion of the observed objects. Other example constraints include the area of a polygon defined by the objects that can be used to constrain the spread of objects within the FOV, constraints on the relative position between the camera and objects that can be used to maintain a desired stand-off distance, the distance of the camera from another object or location (e.g., friendly vehicle or ground station), and approach vectors to the objects.

When multiple constraints are used, there is no guarantee that it will be possible to satisfy or enforce each of them at the same time. Therefore, the priority rank of the constraints is identified, as indicated in block 14. By identifying such priority rank, the most important constraints can be satisfied (if possible) before other less important constraints. For the problem of maintaining multiple moving objects (targets) within a camera FOV, it may make sense to assign the highest priority to the mean constraint, the second highest priority to the variance constraint, and lower priorities to other constraints, if applicable. In some embodiments, the priority rank can be selected by a user. Alternatively, a default ranking can be automatically implemented.

Once the constraints to be enforced have been identified, the parameters for each constraint can be identified, as indicated in block 16. Again, the parameters can be selected by a user or automatically implemented as a default. The parameters used depend upon the underlying constraint. For example, if the constraint is the mean constraint, the parameter can be the location within the camera FOV at which to position the object mean. As mentioned above, the center of the camera FOV is a good candidate for that location when attempting to maintain multiple objects within the FOV. If the constraint is variance constraint, the parameter can be the maximum camera image distance (e.g., in pixels) that any object can move from the object mean (e.g., FOV center).

Once each of the physical constraints, the imposed constraints, the constraint priorities, and the constraint parameters have been identified to or by the system, the system operates to maintain multiple moving objects within the FOV. To do so, the system attempts to enforce each imposed constraint in the order of its priority. In particular, the set of solutions that satisfy the first constraint are determined, the set of solutions that satisfy the second constraint that do not interfere with the first constraint are determined, and so forth until each constraint has been satisfied or a constraint is encountered that cannot be satisfied. As described above in the Discussion of the Control Approach section, this can be accomplished using Equation 28. Blocks 18-28 describe an equivalent process (logical flow) that is performed when solving that equation.

Referring to block 18, the set of solutions, if any, that satisfy the first imposed constraint is determined. As used herein, the term "solutions" relates to camera configurations, which may include one or more of the camera settings, pose, position in three-dimensional space, and velocity. In some embodiments, determining the set of solutions comprises determining the null spaces of the Jacobian matrix associated with the constraint's task function. With reference to decision block 20 of FIG. 2B, if there is no solution to the constraint, the objects cannot all be maintained within the camera FOV and flow for the instant process is terminated. Of course control could be exercised to maintain as many of the objects within the camera FOV as possible. In addition or in alternative, a control routine could be used to recover sight of any lost object. If there are one or more solutions that satisfy the constraint, however, flow proceeds to decision block 22 at which it is determined whether there is another constraint to be satisfied. If not, no other constraints will be satisfied and camera configuration information that identifies a camera configuration that will ensure that the objects are maintained within the camera FOV is output (block 28), for example to an appropriate camera controller. Such configuration information can comprise information as one or more of the camera settings, pose, position, and velocity required to keep all of the objects in the FOV. In some embodiments, the configuration information can comprise a single camera configuration from a group of viable camera configurations. In other embodiments, the configuration information can include every viable camera configuration and the decision as to which to implement can be made by another component entity (e.g., camera controller or camera operator).

Referring again to decision block 22, if there is another imposed constraint to satisfy, flow continues to block 24 at which the set of solutions that satisfy the next constraint and that further do not interfere with any higher priority constraint is determined. In cases in which the solutions are determined by identifying null spaces of Jacobian matrices associated with the constraints, determining the set of solutions for the lower priority constraint comprises identifying the null spaces of the lower priority constraint's Jacobian matrix that lie in the null space of the Jacobian matrix of the higher priority constraint. By doing so, no solution for any constraint will be considered valid unless it is also valid for any preceding higher priority constraint. Referring next to decision block 26, if there is no solution that satisfies the next constraint as well as previous, higher priority constraints, flow proceeds to block 28 at which the previously generated camera configuration information is output. If, however, there is at least one solution that satisfies the next constraint and the preceding constraints, flow returns to decision block 22 at which it is determined whether there is a further imposed constraint to satisfy.

The process described above in relation to blocks 22-26 repeats until either a constraint is encountered that cannot be satisfied while simultaneously satisfying higher-priority constraints or every imposed constraint has been satisfied. In either case, camera configuration information is provided that can be used to control the camera so as to maintain moving objects within the camera FOV.

It is noted that, because the objects within the camera FOV are moving and therefore changing position as a function of time, the process performed in blocks 18-28 is continually performed as long as the objects are tracked so that one or more of the camera settings, pose, and position can be continually adjusted to account for the object movement. By way of example, the system outputs camera configuration information that can be used to adjust the camera at least 30 times a second, which is currently the operating frequency of most standard video cameras.

Simulation Results

Simulations were performed using the proportional controller and PID controller given in Equation 28. Those simulations are described below as Simulations 1 and 2.

1. *Simulation 1: Randomly Moving Points*

In the first simulation, a camera observed eight independent points moving with random acceleration in three dimensions. This simulation mimicked the case of keeping multiple objects in the FOV, such as and airborne camera watching a dispersing crowd or several moving vehicles. In this situation, the features could come from extracted centers of segments in the image. The camera had a resolution of 512x512 . The mean was regulated to $[256,256]^{r}$, i.e., the image center. The variance was regulated to $[100^2, 100^2]^T$, giving a standard deviation of 100 pixels. The simulation was executed for 10 seconds at 30 frames per second.

Results using PID control for mean and variance regulation are presented are presented in FIGs. 3-5. FIG. 3 shows the trajectories of the viewed feature points in the image plane. FIG. 4 shows the error between the current and desired values of the task functions over time. The integral term caused some overshoot in both the mean and variance error regulation, but both task functions were regulated well in the presence of the uncertain disturbance. The perceptibility could not be driven to zero, due to the fact that it had a lower priority than the other tasks, but it was regulated to approximately 0.06. FIG. 5 shows the camera velocity over time, which remains smooth throughout the simulation.

2. *Simulation 2: Two Randomly Moving Objects*

In the second simulation, a camera observed two rigid, square objects. The objects moved independently of each other and the corners of the squares gave eight feature points to track. This simulation mimicked the case of an airborne camera tracking two vehicles. The mean was regulated to the image center. The variance was regulated to $[100^2, 100^2]^r$, i.e. a standard deviation of 100 pixels. The simulation was executed for 10 seconds at 30 frames per second. The square objects moved with sinusoidal velocities along all six degrees of freedom.

The results using the PID controller are shown in FIGs. 6-8. FIG. 6 depicts the trajectory of the viewed feature points in the image plane. The feature points all remained in the FOV. FIG. 7 illustrates the error between the current and desired mean and current and desired variance over time, and it can be seen that tracking error was regulated to about zero for both mean and variance. FIG. 8 shows the camera velocity over time.

Example System

FIG. 9 illustrates an example system 30 that can be used to track multiple moving objects 32 and maintain them within a camera FOV 34. As indicated in that figure, the system 30 generally comprises at least one camera 36 that is communicatively coupled (either with a wired or wireless connection) to a computing system 38. Although the computing system 38 is depicted in FIG. 9 as a conventional computer, the computing system can comprise other configurations. More important than the configuration of the computing system are its capabilities.

FIG. 10 illustrates an example architecture for the computing system 38 shown in FIG. 9. As indicated in FIG. 10, the computing system 38 comprises a processing device 40, memory 42, a user interface 44, and at least one input/output (I/O) device 46, each of which is connected to a local interface 48.

The processing device 40 can comprise a central processing unit (CPU) that controls the overall operation of the computing system 38. The memory 42 includes any one of or a combination of volatile memory elements (e.g., RAM) and nonvolatile memory elements (e.g., hard disk, ROM, etc.) that store code that can be executed by the processing device 40.

The user interface 44 comprises the components with which a user interacts with the computing system 38, such as a keyboard, a mouse, and a computer monitor, or touch-screen monitor. The one or more I/O devices 46 are adapted to facilitate communications with other devices and may include one or more communication components such as a modulator/demodulator (e.g., modem), wireless (e.g., radio frequency (RF)) transceiver, network card, etc.

The memory 42 (i.e., a computer-readable medium) comprises various programs (i.e., logic) including an operating system 50 and an image analysis program 52. The operating system 50 controls the execution of other programs and provides scheduling, input-output control, file and data management, memory management, and communication control and related services. The image analysis program 52 is capable of identifying and tracking target objects within captured images and is configured to determine solutions to the imposed constraints and generate camera configuration information that identifies camera configurations in which objects can be maintained within the camera FOV, in accordance with the foregoing disclosure. The memory 42 further comprises a camera controller 54 that receives the camera configuration information and controls the camera to keep the objects within the FOV. In some embodiments, controlling the camera may include controlling a vehicle to which the camera is mounted to change the position or orientation of the camera.

Various code (i.e., logic) has been described in this disclosure. Such code can be stored on any computer-readable medium for use by or in connection with any computer-related system or method. In the context of this document, a "computer-readable medium" is an electronic, magnetic, optical, or other physical device or means that contains or stores code, such as a computer program, for use by or in connection with a computer-related system or method. The code can be embodied in any computer-readable medium for use by or in connection with an instruction execution system, apparatus, or device, such as a computer-based system, processor-containing system, or other system that can fetch the instructions from the instruction execution system, apparatus, or device and execute the instructions.

The foregoing disclosure is provided for purposes of illustrating, explaining, and describing embodiments of the disclosed inventions. Modifications and adaptations to these embodiments will be apparent to those skilled in the art and may be made without departing from the scope or spirit of this invention. In one such modification, multiple cameras can be networked together so that objects that cannot be maintained within one camera's FOV can be "handed off" to one or more neighboring cameras so that the system can still track the objects. For example, once a first camera determines that an object is about to leave its FOV, the first camera can communicate the position, speed, and direction of travel of the object to a second camera to enable the second camera to add the object to the group of objects it is tracking and, therefore, maintaining within its FOV. With such a system, potentially no object can escape from view of the system.

CLAIMS

We claim:

1. A method for maintaining objects within a camera field of view, the method comprising:

identifying constraints to be enforced, each constraint relating to an attribute of the viewed objects;

identifying a priority rank for the constraints such that more important constraints have a higher priority than less important constraints; and

determining the set of solutions that satisfy the constraints relative to the order of their priority rank such that solutions that satisfy lower ranking constraints are only considered viable if they also satisfy any higher ranking constraints, each solution providing an indication as to how to control the camera to maintain the objects within the camera field of view.

2. The method of claim 1, wherein one constraint relates to the location of a mean position of the objects within the camera field of view.

3. The method of claim 2, further comprising regulating the location of the mean object position so as to be coincident with the center of the camera field of view.

4. The method of claim 1, wherein one constraint relates to the maximum variance permitted for any object from a mean position of the objects.

5. The method of claim 1, wherein one constraint relates to the perceptibility of the objects within the camera field of view.

6. The method of claim 1, wherein determining the set of solutions comprises developing an underdetermined task function for each constraint.

7. The method of claim 6, wherein determining the set of solutions further comprises determining the derivative of each underdetermined task function.

8. The method of claim 7, wherein determining the set of solutions further comprises developing a Jacobian matrix for each underdetermined task function.

9. The method of claim 8, wherein determining the set of solutions further comprises determining null spaces for each Jacobian matrix, the null spaces comprising the solutions satisfy the constraints.

10. The method of claim 9, wherein determining null spaces comprises identifying the null spaces of a Jacobian matrix associated with a lower priority constraint that lie in the null space of a Jacobian matrix associated with a higher priority constraint such that no solution for any constraint is considered valid unless it is also valid for any preceding higher priority constraint.

11. The method of claim 10, wherein the set of solutions is determined using the equation:

$$v_c = L^+(\dot{m}_m + (l - JiJ_m)[m_v + (l - JlJ_v)\dot{m}_p]).$$

12. The method of claim 1, wherein each solution indicates a camera configuration in which the objects will be maintained within the camera field of view.

13. The method of claim 12, wherein the camera configuration includes one or more of the camera's settings, pose, and position, and velocity.

14. The method of claim 1, further comprising outputting one or more of the solutions as camera configuration information to a camera controller that controls the camera configuration.

15. A computer-readable medium that stores instructions that can be executed by a processor for the purpose of maintaining objects within a camera field of view, comprising:

logic configured to identify constraints to be enforced, each constraint relating to an attribute of the viewed objects;

logic configured to identify a priority rank for the constraints such that more important constraints have a higher priority that less important constraints; and

logic configured to determine the set of solutions that satisfy the constraints relative to the order of their priority rank such that solutions that satisfy lower ranking constraints are only considered viable if they also satisfy any higher ranking constraints, each solution providing an indication as to how to control the camera to maintain the objects within the camera field of view.

16. The medium of claim 15, wherein one constraint relates to the location of a mean position of the objects within the camera field of view.

17. The medium of claim 16, further comprising logic configured to regulate the location of the mean object position to be coincident with the center of the camera field of view.

18. The medium of claim 15, wherein one constraint relates to the maximum variance permitted for any object from a mean position of the objects.

19. The medium of claim 15, wherein one constraint relates to the perceptibility of the objects within the camera field of view.

20. The medium of claim 15, wherein the logic configured to determine the set of solutions comprises logic configured to determine the derivative of a underdetermined task function related to each constraint.

21. The medium of claim 20, wherein the logic configured to determine the set of solutions further comprises logic configured to develop a Jacobian matrix for each underdetermined task function.

22. The medium of claim 21, wherein the logic configured to determine the set of solutions further comprises logic configured to determine null spaces for each Jacobian matrix, the null spaces comprising the solutions to each constraint.

23. The medium of claim 22, wherein the logic configured to determine null spaces is configured to identify the null spaces of a Jacobian matrix associated with a lower priority constraint that lie in the null space of a Jacobian matrix associated with a higher priority constraint such that no solution for any constraint is considered valid unless it is also valid for any preceding higher priority constraint.

24. The medium of claim 23, wherein the logic configured to determine the set of solutions is configured to determine the solutions using the equation:

$$V_c = \text{Ü}\ (\dot{m}_m + (l\text{-}JlJ_m)[m, + (l\text{-}JlJ_v)\dot{m}_p]).$$

25. The medium of claim 15, wherein each solution indicates a camera configuration in which the objects will be maintained within the camera field of view.

26. The medium of claim 25, wherein the camera configuration includes one or more of the camera's settings, pose, position, and velocity.

27. A method for maintaining objects within a camera field of view, the method comprising:

developing an underdetermined task function for each of multiple constraints to be enforced, each constraint relating to an attribute of the viewed objects, a first constraint relating to the location of a mean position of the objects within the camera field of view and a second constraint relating to the maximum variance permitted for any object from a mean position of the objects;

assigning a priority rank to each of the constraints such that more important constraints have a higher priority that less important constraints, the first constraint being assigned a higher priority rank than the second constraint;

developing a Jacobian matrix for each task function and therefore each constraint;

determining null spaces for each Jacobian matrix relative to the order of the priority rank of their associated constraints such that the null spaces for each constraint are within the null space of any higher priority constraint, each null space identifying a camera configuration in which the objects will be maintained within the camera field of view; and

outputting camera configuration information to a camera controller that controls the camera configuration.

28. The method of claim 27, further comprising regulating the location of the mean object position to be coincident with the center of the camera field of view.

29. The method of claim 27, wherein a third constraint relates to the perceptibility of the objects within the camera field of view.

30. The method of claim 27, wherein the null spaces for each constraint are determined using the equation:

$$V_c = L^+ (\dot{m}_m + (l - J i J_m)[m_v + (l - J^{\text{ï}} J_v)\dot{m}_p]).$$

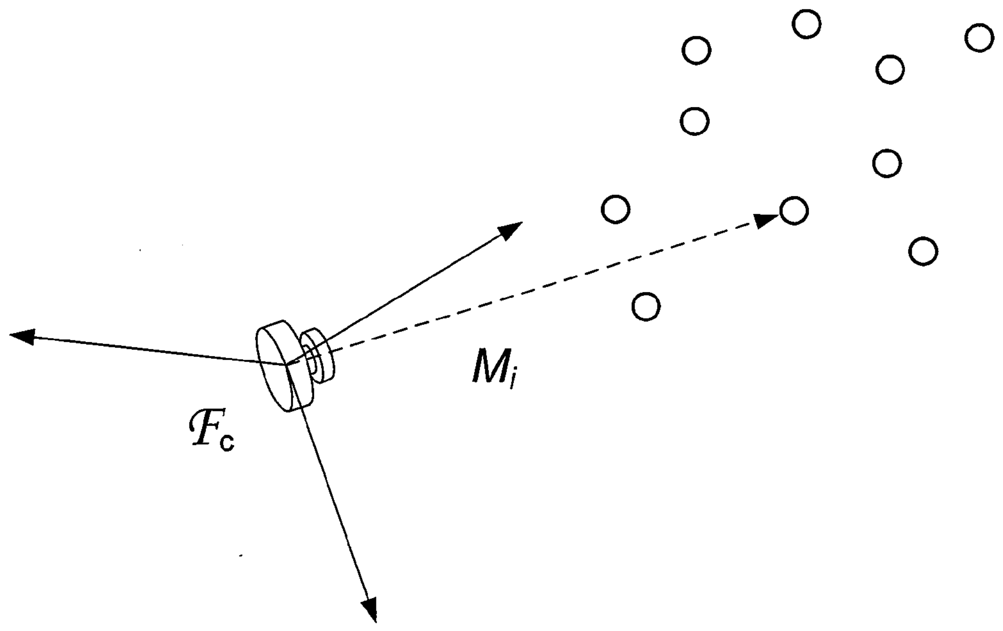

M_i
$\mathcal{F}_c$

FIG. 1

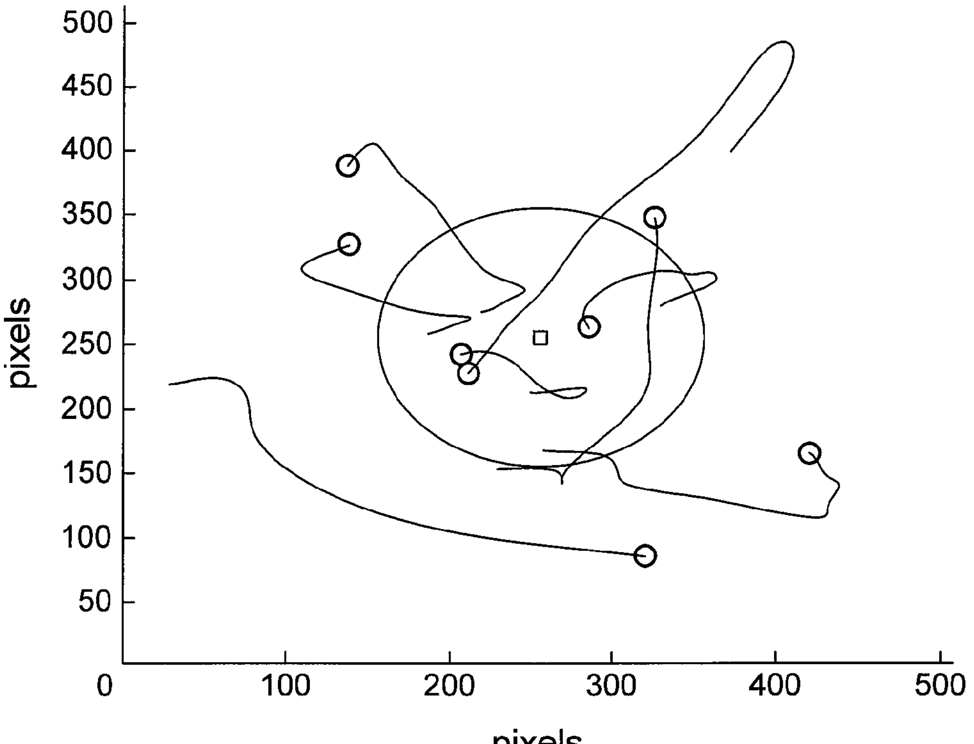

500
450
400
350
300
250
200
150
100
50
0
100
200
300
400
500
pixels
pixels

FIG. 3

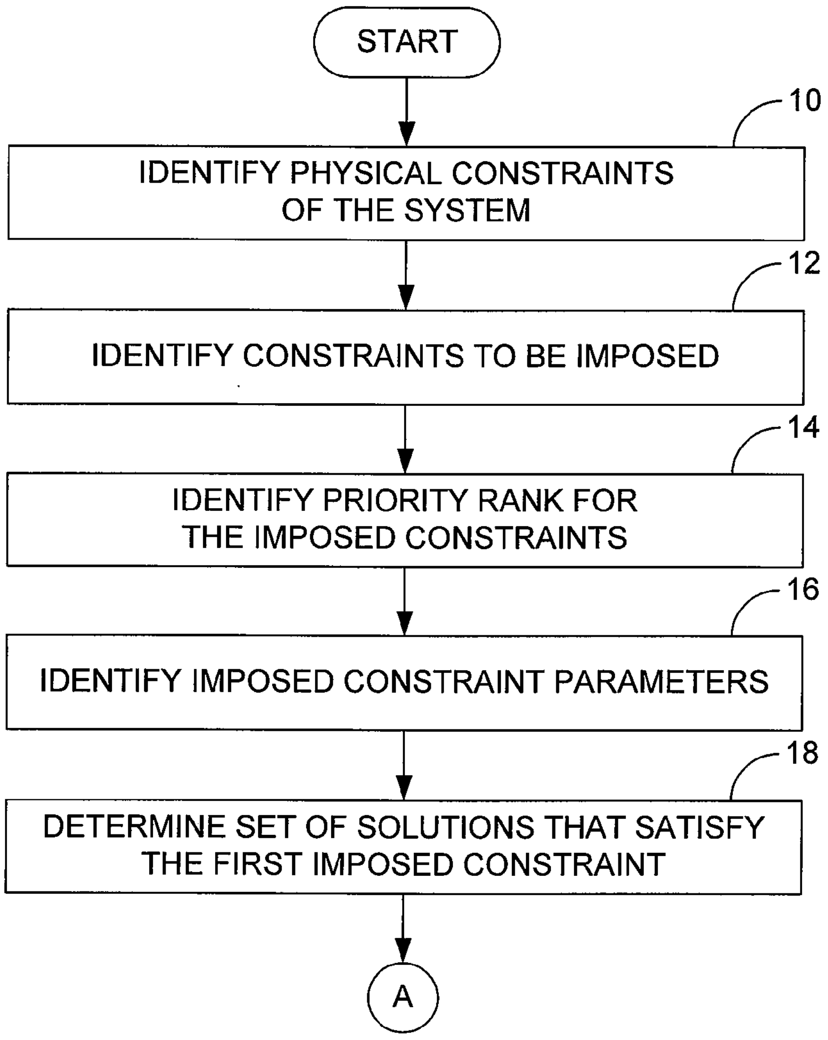

START
10
IDENTIFY PHYSICAL CONSTRAINTS OF THE SYSTEM
12
IDENTIFY CONSTRAINTS TO BE IMPOSED
14
IDENTIFY PRIORITY RANK FOR THE IMPOSED CONSTRAINTS
16
IDENTIFY IMPOSED CONSTRAINT PARAMETERS
18
DETERMINE SET OF SOLUTIONS THAT SATISFY THE FIRST IMPOSED CONSTRAINT
A

FIG. 2A

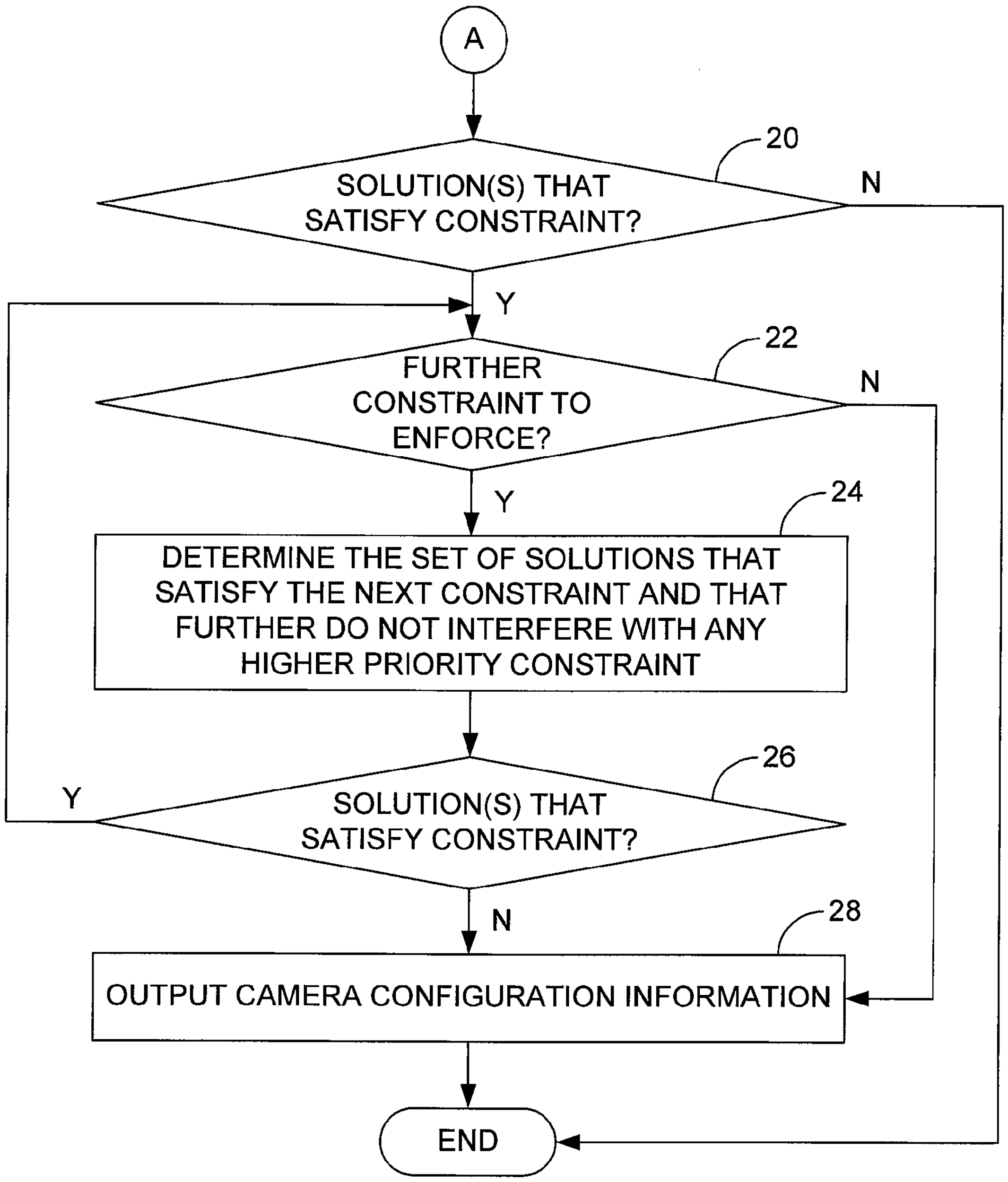
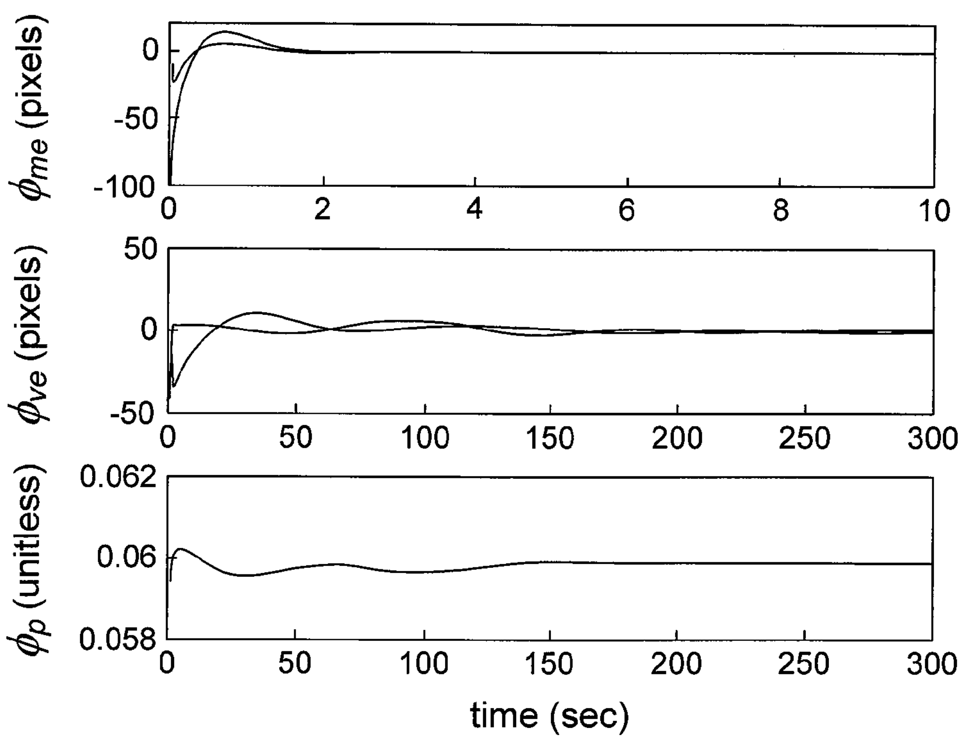
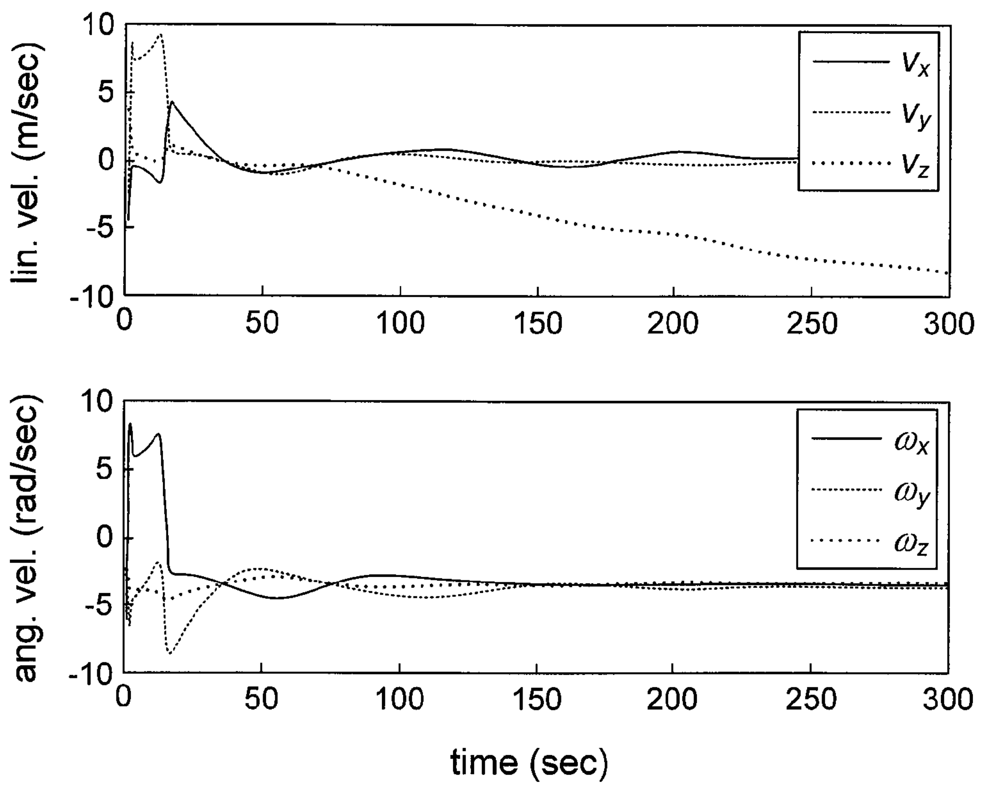

A
SOLUTION(S) THAT SATISFY CONSTRAINT?
20
N
Y
FURTHER CONSTRAINT TO ENFORCE?
22
N
Y
DETERMINE THE SET OF SOLUTIONS THAT SATISFY THE NEXT CONSTRAINT AND THAT FURTHER DO NOT INTERFERE WITH ANY HIGHER PRIORITY CONSTRAINT
24
SOLUTION(S) THAT SATISFY CONSTRAINT?
26
Y
N
OUTPUT CAMERA CONFIGURATION INFORMATION
28
END

FIG. 2B



ϕ_{me} (pixels)
0
-50
-100
0
2
4
6
8
10
ϕ_{ve} (pixels)
50
0
-50
0
50
100
150
200
250
300
ϕ_p (unitless)
0.062
0.06
0.058
0
50
100
150
200
250
300
time (sec)

FIG. 4



lin. vel. (m/sec)
10
5
0
-5
-10
0
50
100
150
200
250
300
v_x
v_y
v_z
ang. vel. (rad/sec)
10
5
0
-5
-10
0
50
100
150
200
250
300
ω_x
ω_y
ω_z
time (sec)

FIG. 5

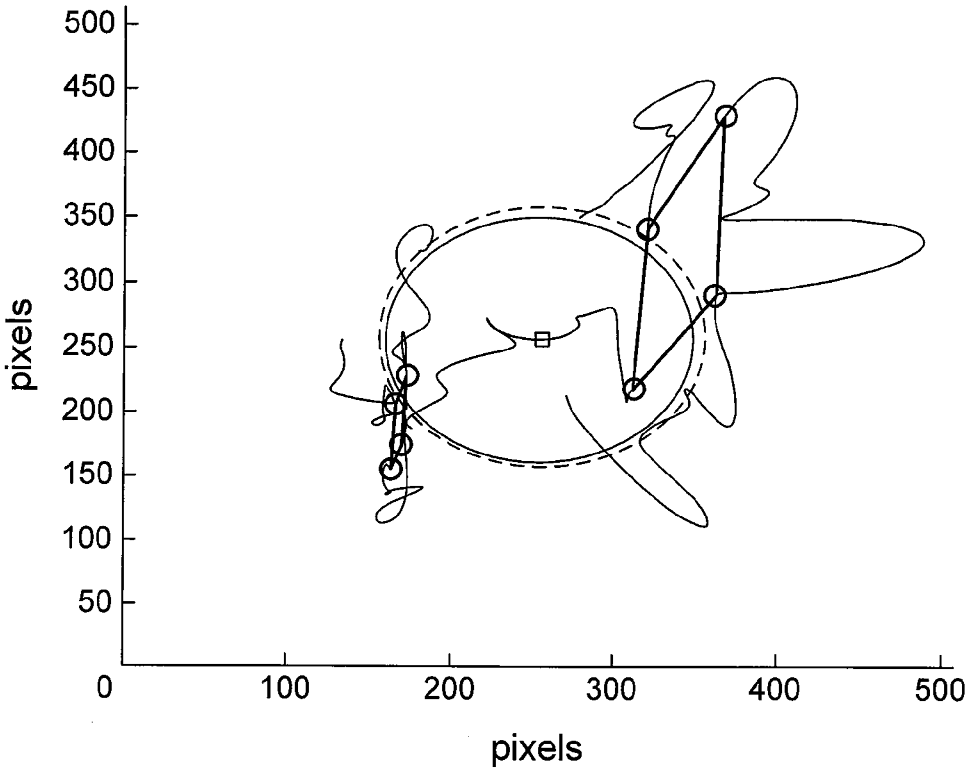

500
450
400
350
300
250
200
150
100
50
pixels
0
100
200
300
400
500
pixels

FIG. 6

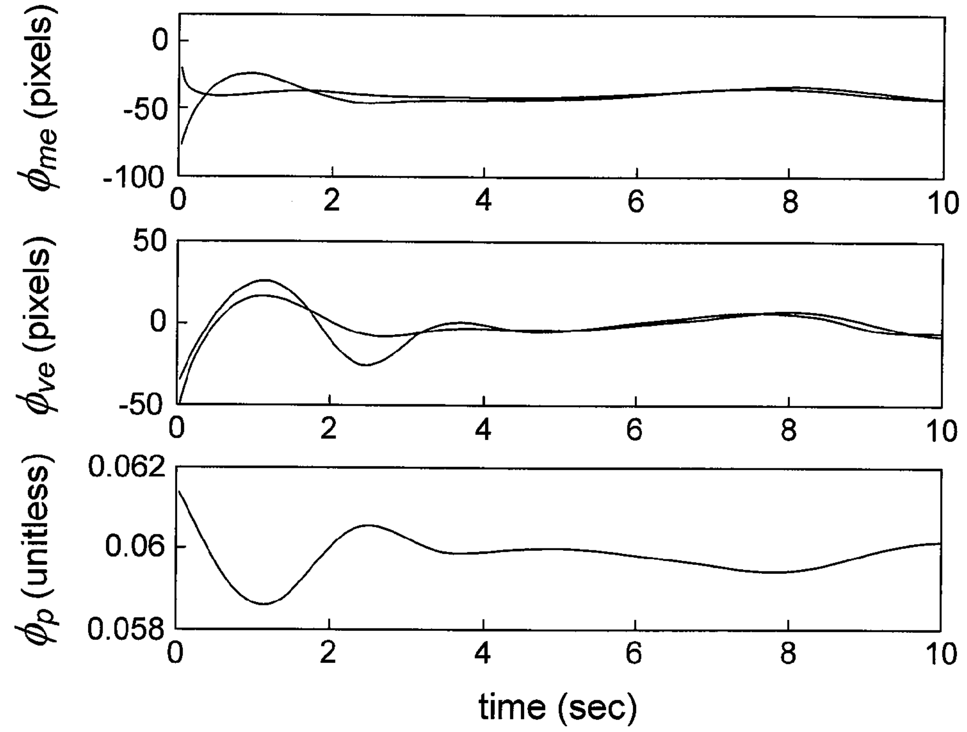
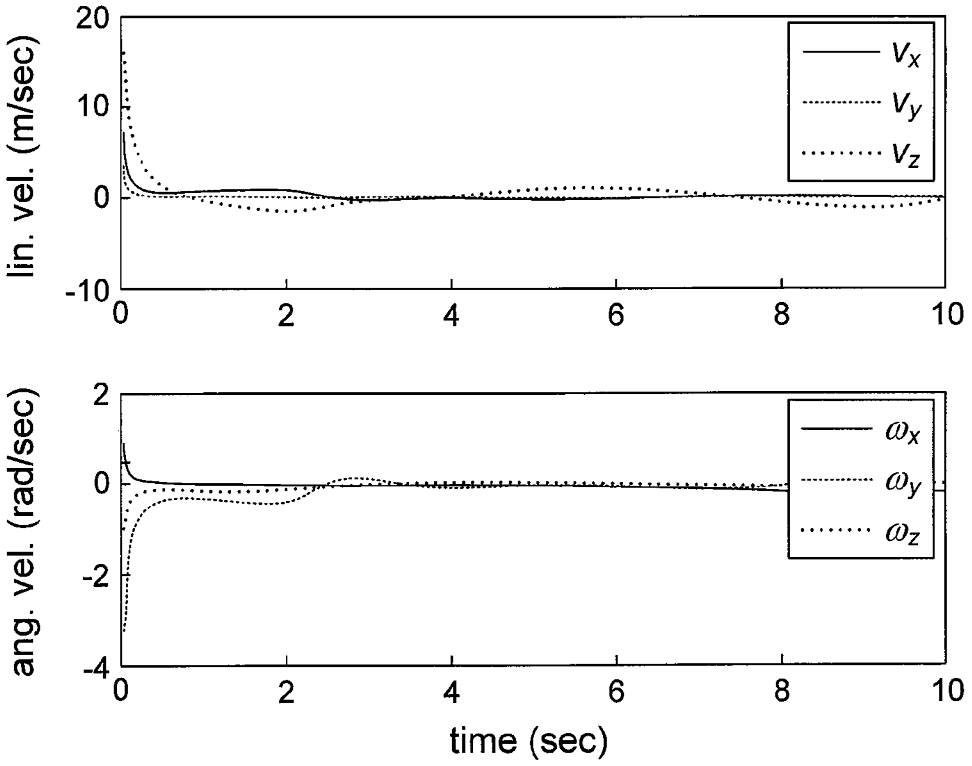
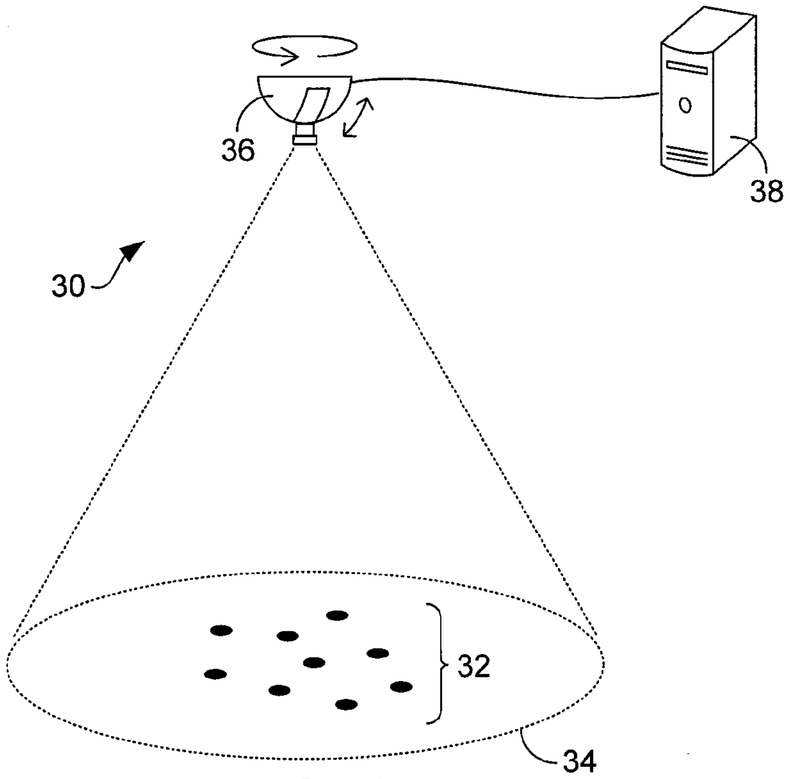

0
-50
-100
ϕ_{me} (pixels)
0
2
4
6
8
10
50
0
-50
ϕ_{ve} (pixels)
0
2
4
6
8
10
0.062
0.06
0.058
ϕ_p (unitless)
0
2
4
6
8
10
time (sec)

FIG. 7



lin. vel. (m/sec)
20
10
0
-10
v_x
v_y
v_z
ang. vel. (rad/sec)
2
0
-2
-4
ω_x
ω_y
ω_z
0
2
4
6
8
10
time (sec)

FIG. 8



36
38
30
32
34

FIG. 9

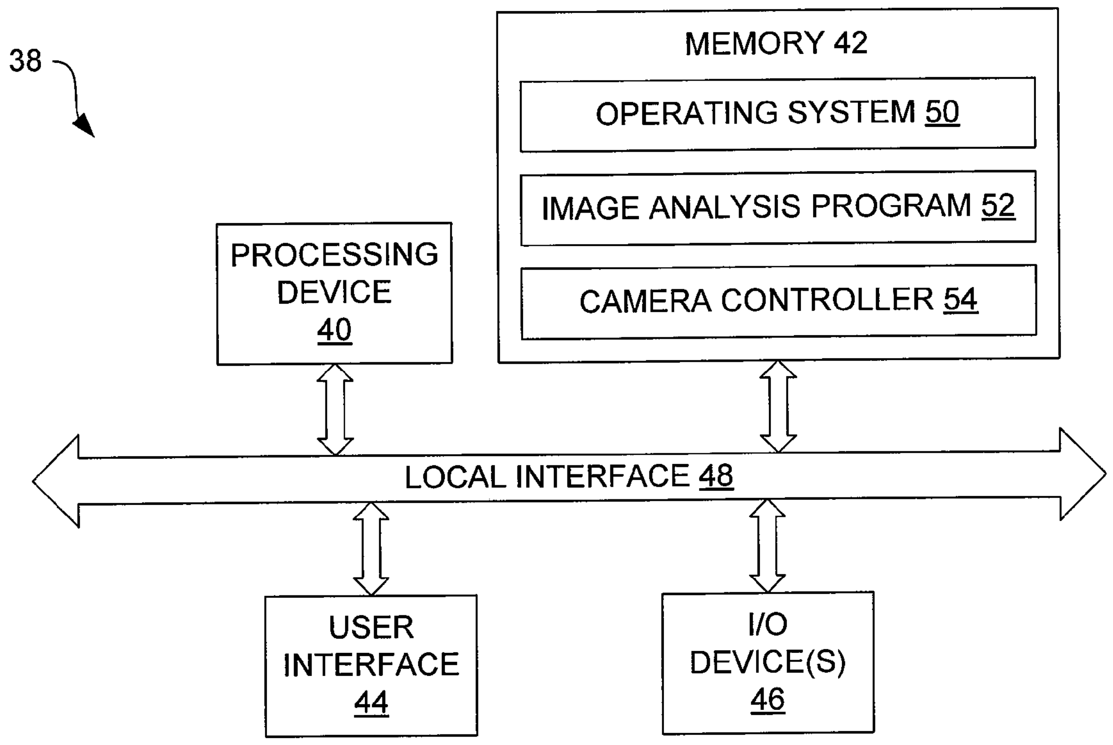

38
MEMORY 42
OPERATING SYSTEM 50
IMAGE ANALYSIS PROGRAM 52
CAMERA CONTROLLER 54
PROCESSING DEVICE 40
LOCAL INTERFACE 48
USER INTERFACE 44
I/O DEVICE(S) 46

FIG. 10