Chapter 10

Intelligent Sliding Surface Design Methods Applied to an IBVS System for Robot Manipulators

Tolga Yüksel

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/63046

Abstract

The controller of an image-based visual servoing (IBVS) system is based on the design of the kinematic velocity controller which guarantees exponentially decreasing feature errors. In fact, this controller is using the sliding surface approach of classical Sliding Mode Control (SMC). In SMC, the system dynamics are taken into consideration and the sliding surface is designed according to the physical limitations and desired convergence time. Different design methods are proposed in the literature using adaptive gain, time variations, nonlinear functions, and intelligent methods like fuzzy logic (FL) and genetic algorithms (GA). In this study, five different sliding surface designs with analytical and intelligent methods are modified and applied to an IBVS system to expand these designs to visually guided robot manipulators. The design methods are selected by their convenience and applicability to these types of manipulator systems. To show the performance of the design methods, an IBVS system with six-DOF manipulator is simulated using MATLAB *Simulink, Robotics Toolbox, Machine Vision Toolbox,* and *Fuzzy Logic Toolbox*. A comparison of these design methods according to convergence time, error cost function, defined parameters, and motion characteristics is given.

Keywords: Sliding surface, visual servoing, robot manipulators, fuzzy logic, Simulink

1. Introduction

Visual servoing (VS) uses the camera image to control the motion of a robot, and it needs points or important pixels named as features in an image. In VS, image plane is used to define these k feature points and the coordinates of these features in image plane form, the vector s. Error signal vector is derived from the difference between the desired vector s^* and s, and these signals are used to obtain velocity control law. By using these definitions, VS methods are mentioned

in two main titles. Image-based visual servoing (IBVS) uses s vector obtained from the image directly but position-based visual servoing (PBVS) needs s obtained from 3D parameter estimations of image and robot pose [1]. This advantage and robustness against depth estimation errors make IBVS more attractive and applicable in VS. Here, it must be noted that featureless visual servoing approaches like kernel-based or luminance-based methods are becoming more popular [2, 3], but feature-based methods will continue their royalty. Besides these features, the configuration of the camera and the end effector should be taken into consideration in VS. Eye-in-hand configuration, as the most popular configuration in VS, is chosen for this study. This configuration is also a step of VS for unmanned air vehicles (UAV).

Studies on IBVS mostly focus on different feature extraction methods [4], different camera geometries and types [5], hybrid VS methods [6] or other problems of VS like singularity avoidance or field of view (FOV) keeping [7] but the controller design doesn't draw too much attention because the linear velocity controller design approach of VS is assumed as the sufficient controller as the errors are decreasing exponentially. Besides this common sense, other performance parameters like convergence time, velocity limits, and error magnitudes are actually real-time metrics. For robot applications in a production line, these parameters become dominant to speed up process to increase accuracy and repeatability. Some other control approaches like visual predictive control [8] fusing predictive control with VS are promising but the controller should be applicable in real-time.

To design a new IBVS controller, the velocity controller design approach of IBVS should be examined. As explained in details in the following section, the linear controller of IBVS is using sliding surface design of sliding mode control (SMC). In SMC, two design steps are defined. The first one is designing a sliding surface that represent desired stable states of the system, and the second one is designing a control law that guarantee states reaching to the sliding surface and sliding on this surface [9]. A sliding slope is defined in SMC and this slope is named as gain in IBVS. The rule of thumb of an appropriate slope in SMC is choosing a slope small enough in order not to exceed control limits and conversely, choosing a slope big enough to reach the sliding slope faster and slide on this surface faster. This is interpreted by IBVS as choosing gain small enough in order not to exceed velocity limits and choosing gain big enough to converge faster.

From initial conditions to convergence, there are two modes of state trajectories in SMC. In reaching mode, the states are not on the sliding surface and they try to reach this surface. In sliding mode, they are on the sliding surface and they try to converge to zero. IBVS and SMC uses fixed gain-sliding slope but alternatively, in the literature, there are many different sliding surface designs which are important from the practical point of view and for high performance. Two main approaches are mentioned in the literature of sliding surface design: linear and nonlinear [10]. Although there are numerous designs for linear surfaces, it is very hard to obtain a nonlinear surface using linear surfaces for high performance, and it is hard to define the parameters of these designs. There are not too many options and the parameters are easily defined for nonlinear designs, but the magnitude of control signals should be observed to avoid saturation [10, 11]. The titles of the most common linear designs are linear constant surface [9], integral sliding surface [12], linear varying (rotating and shifting) sliding surface

http://dx.doi.org/10.5772/63046

[13], and linear time-varying sliding surface [14] designs. Nonlinear designs are more popular and titled as constant nonlinear sliding surface [15], higher order sliding surface [16], fractional sliding surface [17], terminal sliding surface [18], nonlinear sliding surface with nonlinear functions [19], and nonlinear moving sliding and terminal sliding surface designs [14]. Furthermore, intelligent methods like fuzzy logic (FL) and genetic algorithms (GA) are also deployed for parameter assignment of linear and nonlinear surface designs [20]. Detailed reviews on sliding surface designs can be found in [11, 14, 20].

Most studies on VS focus on image processing-feature extraction part of vision, but control part is still open to old and new approaches. In this study, five different sliding surface designs with analytical and intelligent methods are modified and applied to an IBVS system to expand these designs to visually guided robot manipulators. The design methods are selected according to their relevance and applicability to these manipulator systems. In the first design, linear varying sliding surface with FL is assumed. Error and error derivative are used as inputs of FL and to define linguistic rules of FL, the effects of gain on IBVS and experience on IBVS are used. In the second design, integral sliding mode is assumed and the sliding parameter of integral term is tuned using FL. In the third design, time-optimal varying sliding surface design with constant acceleration is assumed. A time interval is defined for this design and linear sliding mode is modified according to this time interval. In the fourth design, a nonlinear tangent hyperbolic function with a width parameter function is used to define a sliding surface. In the fifth design, nonlinear time-varying sliding surface is assumed. The surface is obtained by the product of initial error-error derivatives and an exponential time-varying term. To show the performance of these designs, an error cost function is defined as in [19]. An IBVS system with six-DOF manipulator is simulated and the designs are tested on this system. A comparison of these design methods according to convergence time, error cost function, defined parameters, and motion characteristics is given.

The paper is organized as follows. In the following section, IBVS and used sliding surface designs with modifications and adaptations to IBVS are explained briefly. In Section 3, simulation results for classical IBVS and IBVS with five sliding surface designs are shown. In the last section, an overall comparison of these designs is given and conclusions of the study and future goals are discussed.

2. A review on IBVS and sliding surface designs

In this section, a review on IBVS and sliding surface designs used in the study is given. The main objective of IBVS, as all VS approaches, is to minimize errors derived from k feature point vector s (1)

$$e(t) = s - s^* \tag{1}$$

where $e(t) \in \mathbb{R}^k$ is the error vector and s^* is the desired features for fixed-motionless feature points with zero derivatives. Changes in s depend only on camera motion. Furthermore, IBVS

assumes that the camera is attached to the end effector of a six DOF arm as eye-in-hand configuration and $k \geq 6$.

In the classical IBVS, a point $P = (X, Y, Z)$ in 3D camera frame is defined in image plane with a point p using perspective projection and the velocity of P relative to camera frame is given in terms of linear velocity V and angular velocity Ω

$$\dot{P} = -\Omega \times P - V \tag{2}$$

This velocity is used to obtain transformation between the velocity of the coordinates of perspective projection point (u, v) with a focal length λ_f and the linear and angular velocities of point P. This transformation is defined as below:

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} -\frac{\lambda_f}{Z} & 0 & \frac{u}{Z} & \frac{uv}{\lambda_f} & -\frac{\lambda_f^2 + u^2}{\lambda_f} & v \\ 0 & -\frac{\lambda_f}{Z} & \frac{v}{Z} & \frac{\lambda_f^2 + v^2}{\lambda_f} & -\frac{uv}{\lambda_f} & -u \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ w_x \\ w_y \\ w_z \end{bmatrix} = \dot{s} = L_s.\upsilon_c \tag{3}$$

where υ_c (ν_c, w_c) is the vector of camera's linear and angular velocities in reference coordinate frame. The transformation matrix, $L_s \in \mathbf{R}^{k \times 6}$ is known as the interaction matrix or commonly called image or feature Jacobian matrix. Z is the depth of P and it is hard to obtain the actual value of this term in practice. In most approaches, the estimation value for Z is used and the interaction matrix becomes an estimated interaction matrix $\hat{L}_s$.

By using (1) and (3), relation between error and velocity is obtained as $\dot{s}^*$ is zero

$$\dot{e} = L_s.\upsilon_c \tag{4}$$

IBVS tries to decrease the error exponentially by using the differential equation

$$\dot{e} + \lambda.e = 0 \tag{5}$$

From another aspect, this differential equation is the main equation used by classical SMC to define a sliding surface. Classical SMC defines a surface in terms of error and error derivatives according to the system degree and tries to hold the system states in this surface by equalizing this surface to zero. This definition strongly connects SMC and IBVS. IBVS proposes a

http://dx.doi.org/10.5772/63046

kinematic velocity controller and the velocity signals are the control signals. Again, by using (4) and (5), velocity signals is defined as

$$\begin{aligned} &\dot{e}+\lambda.e=0\rightarrow\dot{e}=-\lambda.e\rightarrow\hat{L}_s.\upsilon_c \\ &=-\lambda.e\rightarrow\upsilon_c=-\lambda.\hat{L}_s^{+}.e \end{aligned} \tag{6}$$

where, $\hat{L}_s^{+}$ is the pseudo-inverse matrix of estimated interaction matrix in cases of non-square matrix and λ is the gain value. Here, it must be noted that this gain value can be in matrix form to define different gain values for each velocity term. This velocity controller is a proportional controller and detailed stability analysis of this controller for IBVS can be found in [1].

2.1. Linear varying sliding surface with fuzzy logic

Varying sliding slope is a common approach in sliding surface designs to achieve desired performance without exceeding velocity limits. The only design parameter in (5) and (6) is the sliding slope, and an on-line parameter tuning algorithm can be proposed by using varying sliding slope approach. Instead of an analytical approach which proposes different parameters to tune a parameter, and which has to be tuned carefully to allow soft parameter variation, FL can be a good candidate not only to avoid tuning twice but also to include linguistic definitions and user experience in the design [21]. IBVS velocity controller in (6) can be modified as in (7) and the block diagram of IBVS with linear varying sliding surface design using fuzzy logic is shown in **Figure 1**.

$$\upsilon_c=-\lambda_{FL}\left(e,\dot{e}\right).\hat{L}_s^{+}.e \tag{7}$$

Figure 1. IBVS with linear varying sliding surface design using fuzzy logic.

In **Figure 1**, it must be noted that limiters should be placed before FL block to restrict λ_{FL}. Membership functions and FL type should be chosen wisely according to desired varying λ_{FL} surface, which is a function of error and error derivative. Furthermore, the rulebase should represent the behavior of classical IBVS under varying gain. An example rule from FL rulebase is defined as follows:

$$\text{IF } e \text{ is HIGH and } \dot{e} \text{ is HIGH, THEN } \lambda_{FL} \text{ is LOW} \tag{8}$$

2.2. Integral sliding surface with fuzzy logic

The classical linear sliding surface design in (5) is a PD-type sliding surface and an integral term can be added to this design to increase tracking performance [12, 22]. This term can be active for all error trajectories or only when the errors are in predefined bounds. Integral sliding surface is given below:

$$\lambda_p.e + \lambda_I.\int e.dt + \lambda_D.\dot{e} = 0 \tag{9}$$

Finding appropriate gain values (λ_P, λ_I, λ_D) for this surface design is the main problem. Again, FL can be applied to this design as in [23]. In this study, only gain value for integral term (λ_I) is assigned using FL as a function of$e$. (9) is reformulated for IBVS in (10) and the block diagram of IBVS with integral surface design using fuzzy logic is shown in **Figure 2**.

$$\upsilon_c = -\hat{L}_s^{+}.\left(\lambda_p.e + \lambda_{I(FL)}(e).\int e.dt\right) \tag{10}$$

Figure 2. IBVS with integral surface design using fuzzy logic.

2.3. Time-varying sliding surface design with constant acceleration

As mentioned in [14], trade-off between short sliding slope reaching phase and slower responses in sliding surfaces can be bested by time-varying sliding slopes. A time-varying sliding slope term, which is defined as a function of time is added to classical sliding surface design and this term is only active until a predefined time instant. In his study, Bartoszewicz designed two different time-varying sliding surfaces, constant-acceleration, and constant-velocity, according to time dependence [14]. Constant-acceleration time-varying sliding surface offers a faster convergence speed to classical sliding surface and this design is chosen in this study. Constant-acceleration time-varying sliding surface design is given in (11) and definitions for the design parameters are given in (12), respectively

http://dx.doi.org/10.5772/63046

$$\dot{e} + \lambda.e + \left\{ \begin{matrix} A.t^2 + B.t + C & \text{for } t \le T \\ 0 & \text{for } t > T \end{matrix} \right\} = 0 \tag{11}$$

$$\begin{aligned} A &= -\frac{\dot{e}(0) + \beta.e(0)}{T^2} \\ B &= 2.\frac{\dot{e}(0) + \beta.e(0)}{T} \\ C &= -\dot{e}(0) - \beta.e(0) \end{aligned} \tag{12}$$

where β is fixed initial error constant and T is time instant for constant acceleration.

The velocity controller and the block diagram of IBVS with time-varying sliding surface design is shown in (13) and **Figure 3** with sample/hold (S/H) units, respectively

$$\upsilon_c = -\hat{L}_s^{+}.\left(\lambda.e + \left\{ \begin{matrix} A.t^2 + B.t + C & \text{for } t \le T \\ 0 & \text{for } t > T \end{matrix} \right\} \right) \tag{13}$$

Figure 3. IBVS with time-varying sliding surface design with constant acceleration.

2.4. Nonlinear sliding surface design with tangent hyperbolic function

Sliding surfaces in the first three designs are linear, which means that error and error derivative will pursue a constant sliding slope. In fact, linear varying or time-varying sliding surfaces reveals piecewise linear surfaces. Instead of using these linear methods, a nonlinear function can be assigned as a sliding surface function [15]. Tangent hyperbolic function can be a good candidate as a nonlinear function [11] and it is preferred in this study. The definition of nonlinear sliding surface design with tangent hyperbolic function is given in (14)

$$\dot{e} + w_p.tanh\left(c_1.e\right) = 0 \tag{14}$$

where w_p is the sliding magnitude parameter which represents convergence speed and c_1 is the sliding slope of the nonlinear surface. These parameters can be assigned constant but to reach sliding surface faster, w_p is chosen as a function of error. The velocity output and block diagram of IBVS with nonlinear sliding surface design using tangent hyperbolic function are shown in (15) and **Figure 4**, respectively.

$$\upsilon_c = -\hat{L}_s^{+}.w_p\left(e\right).tanh\left(c_1.e\right) \tag{15}$$

Figure 4. IBVS with nonlinear sliding surface design using tangent hyperbolic function.

2.5. Nonlinear time-varying sliding surface

As an alternative to nonlinear constant functions, nonlinear time-varying surfaces can be designed to shorten reaching phase. Exponential time-varying functions [14, 19, 24, 25] are proposed in the literature as nonlinear time-varying sliding surface functions. In this study, exponential nonlinear time-varying sliding surface design in (19) is chosen and it is given in (16)

$$\dot{e} + \lambda.e - \left[\dot{e}\left(0\right) + \lambda.e\left(0\right)\right].e^{-k.t} = 0 \tag{16}$$

where λ and k are design parameters of sliding slope and time constant of convergence. This function also includes initial values of error and error derivative and this provides zero initial velocity signals for IBVS. The velocity output and block diagram of IBVS with nonlinear time-varying sliding surface are shown in (17) and **Figure 5**, respectively.

$$\upsilon_c = -\hat{L}_s^{+}\left(\lambda.e - \left[\dot{e}\left(0\right) + \lambda.e\left(0\right)\right].e^{-k.t}\right) \tag{17}$$

http://dx.doi.org/10.5772/63046

Figure 5. IBVS with nonlinear time-varying sliding surface design.

3. Simulation results

In this study, an IBVS system with intelligent sliding surfaces is simulated using MATLAB *Simulink, Robotics Toolbox, Machine Vision Toolbox* and *Fuzzy Logic Toolbox* [26]. As the robot manipulator platform, a six DOF Puma560 arm kinematic model is used in the simulations [27]. Here, it must be noted that inner loop robot manipulator dynamics and control can be fused with IBVS control as in [28], but this study neglects dynamics of the manipulator systems and assumes the IBVS controller as a kinematic velocity controller.

Some assumptions in the study should be given before simulation results:

Assumption 1: It is assumed that there is no transformation between the end effector and the camera mounted on the end effector with eye-in-hand configuration.

Assumption 2: All feature points are collinear.

The camera parameters are defined as follows: the resolution of the camera is 1024×1024, the principal point is (512, 512), and the focal length is 8 mm. The feature points in Cartesian coordinates are defined as P^* using four fixed corner points of a square with 0.5 m side length and $s^* \in \mathbb{R}^{2\times4}$. The center of these points should collide with the principal point as s^*. P^* and s^* in matrix form for feature points are given below:

$$P^*(X,Y,Z) = \begin{bmatrix} 0.25 & 0.25 & -0.25 & -2.5 \\ -0.25 & 0.25 & 0.25 & -0.25 \\ 2.5 & 2.5 & 2.5 & 2.5 \end{bmatrix}$$
$$s^*(u,v) = \begin{bmatrix} 612 & 412 & 412 & 612 \\ 412 & 412 & 612 & 612 \end{bmatrix} \tag{18}$$

The estimated depth value for $\hat{L}_s$ is assumed as 2 m and this value can be considered as a good depth estimation. In the following subsections, details and parameters of classical IBVS and five IBVS sliding surface designs are given and the results are illustrated. To show the performance and the robustness of the designs, random w disturbance with -1< w <1 and zero mean is added to each feature point. As an illustration of the simulated IBVS systems, the block diagram of the closed loop IBVS system with sliding surface design and its *Simulink* model is shown in **Figure 6** and **Figure 7**, respectively. For all these simulations, initial joint angle vector q_0 is given below and the desired pose of Puma 560 with P^* in 3D is shown in **Figure 8**.

$$q_0 = \begin{bmatrix} 0 & \pi/4 & \pi & \pi/10 & \pi/4 & -\pi/4 \end{bmatrix} \text{ rad.} \tag{19}$$

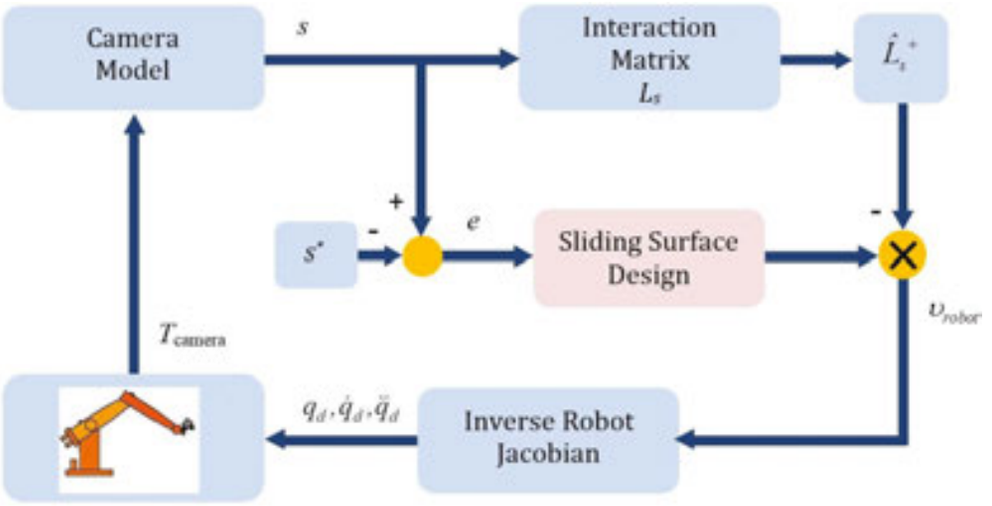

Figure 6. The block diagram of the simulated IBVS systems with sliding surface design.

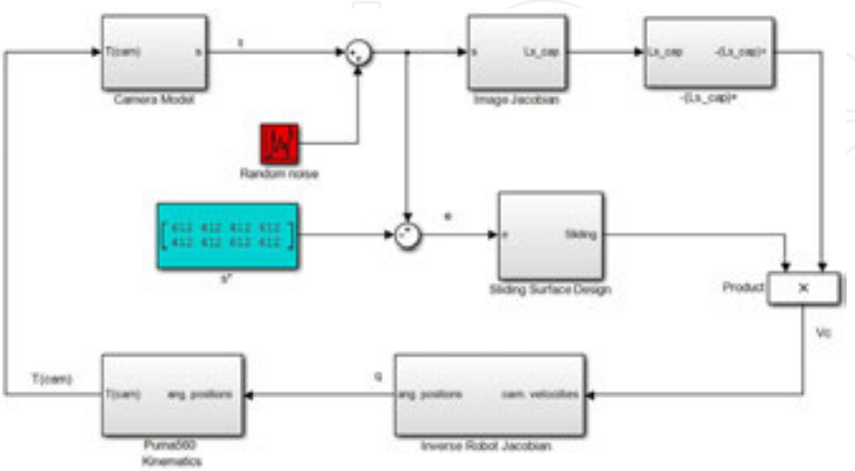

Figure 7. The *Simulink* model of the simulated IBVS systems with sliding surface design.

http://dx.doi.org/10.5772/63046

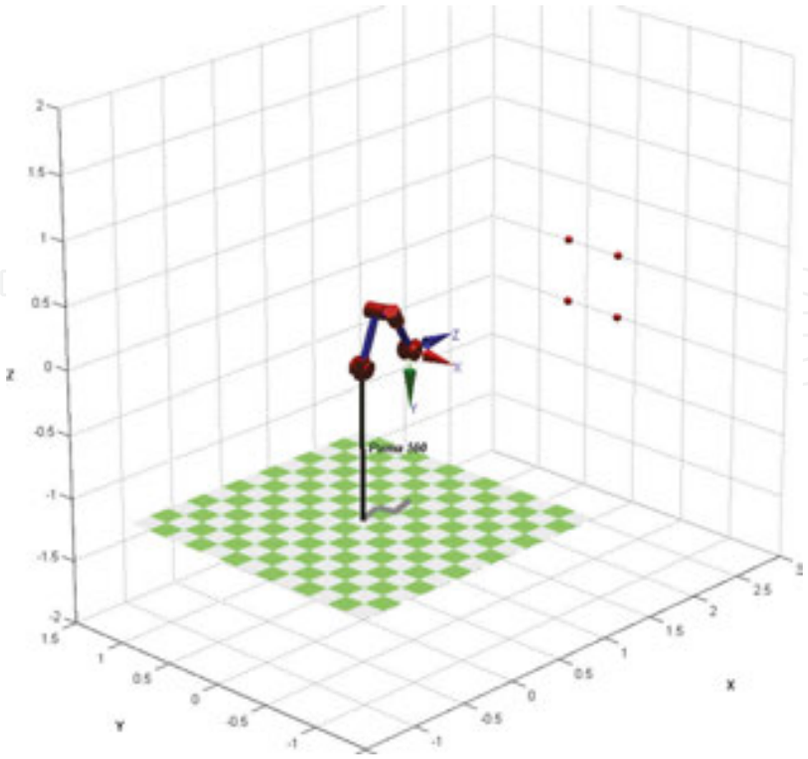

Figure 8. The desired pose of Puma 560 with P^* as red balls.

The main performance criterions for an IBVS system are convergence time (t_{conv}) and velocity limits. In this study, an error cost function is also added to these criterions to show the performance of each design and it is given in (20)

$$J(e) = \sum_{i=1}^{2\times k} \int_0^{t_{conv}} |e_i(t)|.dt \tag{20}$$

where e_i is the error of each feature and i is the index of feature error. Besides these performance numbers, the parameters which have to be assigned for each design are another metric for a closed loop system. These parameters are also discussed in each design. Furthermore, feature motions in image plane are examined to discuss FOV keeping capability for future studies. The inside of the sliding surface block for each design is also given.

3.1. Case 1: Classical IBVS with fixed sliding slope

To make a comparison between each design, a classical IBVS system with fixed sliding slope is assumed as in (5) and (6). The fixed value for λ is chosen as 0.5. Inside the *Simulink* block of the sliding surface design for this case is shown in **Figure 9**.

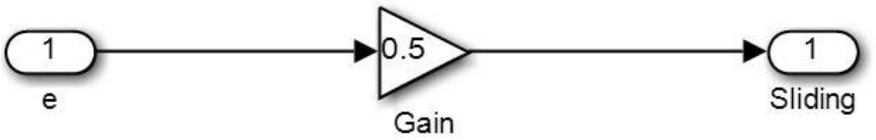

Figure 9. The *Simulink* block of sliding surface for Case 1.

The feature motions in image plane are shown in **Figure 10.a** with initial feature points in red circles and target feature points in blue circles. The velocity signals of the end effector are shown in **Figure 10.b**, and the error signals are shown in Figure 10.c, respectively.

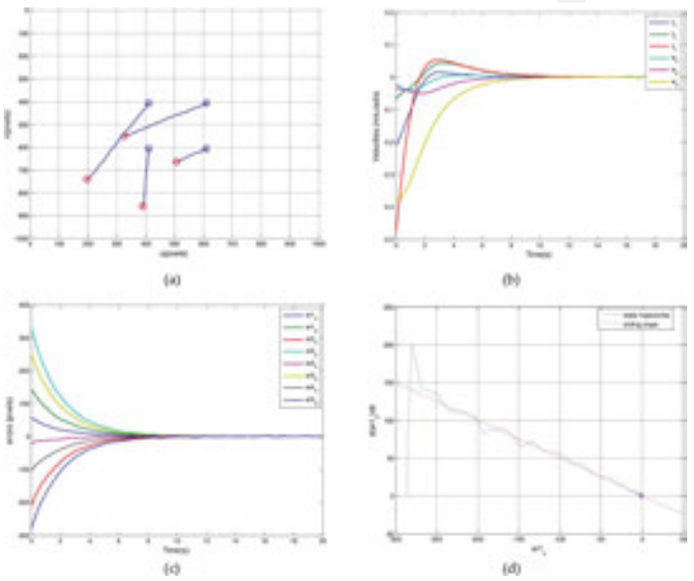

Figure 10. The results for Case 1. (a) Feature motions in image plane (b) Velocities of the end effector (c) Feature errors (d) Error vs. error derivative trajectory.

As shown in **Figure 10.a-b**, the paths of the feature points are linear and the errors are exponentially decreasing as expected from classical IBVS [1]. As the convergence time, the time instant when all absolute errors decreases under 1 pixel is assumed and it is 12.89 s as shown in **Figure 10.c**. The error cost for Case 1 is 2482. As an example of fixed sliding slope, error and error derivative trajectory for the first feature in u coordinate is shown in **Figure 10.d**.

3.2. Case 2: IBVS with linear varying sliding surface using fuzzy logic

The choice of an appropriate λ plays a critical role in the performance of the proportional velocity controller in (6). As the second design, linear sliding slope design with varying λ for fast convergence and low velocity profile is aimed, and assigning the slope by using fuzzy logic that is an approach in fuzzy sliding mode is proposed.

IBVS derives different error signals for each feature. FL needs error and error derivative for an appropriate λ but it is very hard to decouple the error signals of each feature to associate

http://dx.doi.org/10.5772/63046

with velocity signals. To avoid this problem, Euclidian norm values of the error and error derivative vectors are used as the inputs of FL in the design of the sliding surface. Inside the *Simulink* block of the sliding surface design for this case is shown in **Figure 11**. Fuzzy Logic Controller block in **Figure 11** is implemented MATLAB *Fuzzy Logic Toolbox*.

Figure 11. The *Simulink* block of sliding surface for Case 2.

The type of FL unit is Mamdani and the membership functions of the error, the error derivative norm inputs and output λ_{FL} are shown in **Figure 12**. For a better softened varying λ_{FL}, the membership functions are chosen as gaussian and gaussian-bell curve. For each input and output, three membership functions are defined. The rulebase for FL is defined using nine rules and this rulebase is given in **Table 1**. The rules are defined according to the approach as in (8). The FL surface for nonlinear mapping between the inputs and the output is demonstrated in **Figure 13**. Here, it must be noted that λ_{FL} varies between 0.51 and 1.85 as shown in **Figure 13**.

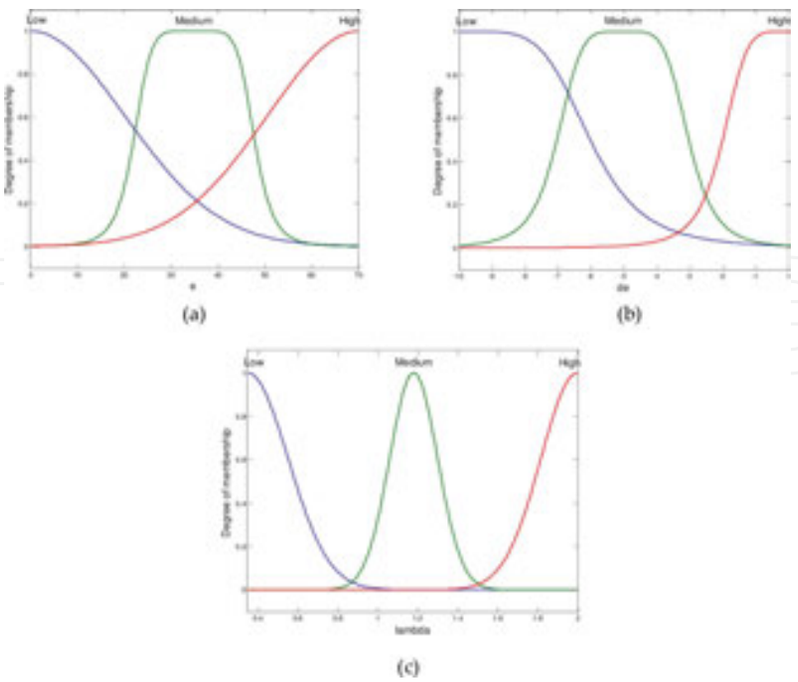

Figure 12. FL membership functions for λ_{FL} (a) Input *e* (b) Input d*e*/d*t* (c) Output λ_{FL} .

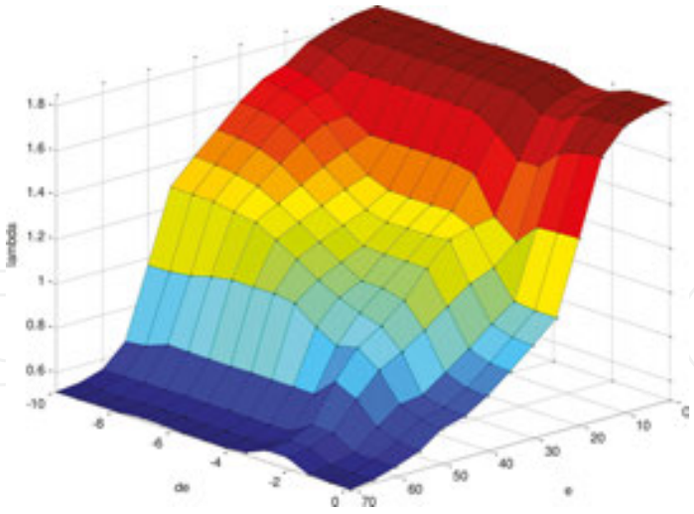

Figure 13. FL surface for λ_{FL} .

Figure 14. The results for Case 2. (a) Feature motions (b) Velocities of the end effector (c) Feature errors (d) λ_{FL} (e) Error vs. error derivative trajectory.

http://dx.doi.org/10.5772/63046

e / $\dot{e}$	Low	Medium	High
Low	High	High	High
Medium	High	Medium	Low
High	Low	Low	Low

Table 1. FL rulebase for linear varying sliding surface.

The feature motions in image plane are shown in **Figure 14.a**, the velocity signals of the end effector are shown in **Figure 14.b**, and the error signals are shown in **Figure 14.c**, respectively.

The varying sliding slope λ_{FL} is shown in **Figure 14.d** and it remains constant and low since error and error derivative norms are high. Time interval for this steady condition is between 0 and 3.98 s. Within this time interval, the slope is fixed and error and error derivative follows this slope as shown in **Figure 14.e**. Then, FL unit is activated by these norms and varying sliding slope shows a nonlinear characteristic without discontinuity. Again, it remains constant and high after norms are small with small fluctuations after 6.15 s. These fluctuations are a consequence of noise added to the features.

The velocity profiles dwell in velocity limits and increases after 3.98 s as shown in **Figure 14.b**. The errors decreases rapidly after varying sliding slope as shown in **Figure 14.c** and error and error derivative try to follow varying sliding slope after 3.98 s. The convergence time is 7.35 s and the error cost is 2591 for Case 2. In **Figure 14.a**, it can be seen that the feature motions are linear, which means that this sliding approach doesn't affect motion characteristics of IBVS.

3.3. Case 3: IBVS with integral sliding surface using fuzzy logic

IBVS with integral sliding surface in (10) is assumed as Case 3 and λ_P is chosen as 0.5. FL is used for λ_I assignment. Inside the *Simulink* block of the sliding surface design for this case is shown in **Figure 15**.

Figure 15. The *Simulink* block of sliding surface for Case 3.

FL unit uses error as input, two membership functions are defined for this input and the FL surface curve is shown in **Figure 16**. Here, it must be noted that limits of this curve, thus λ_I, can be changed but this affect velocity limits.

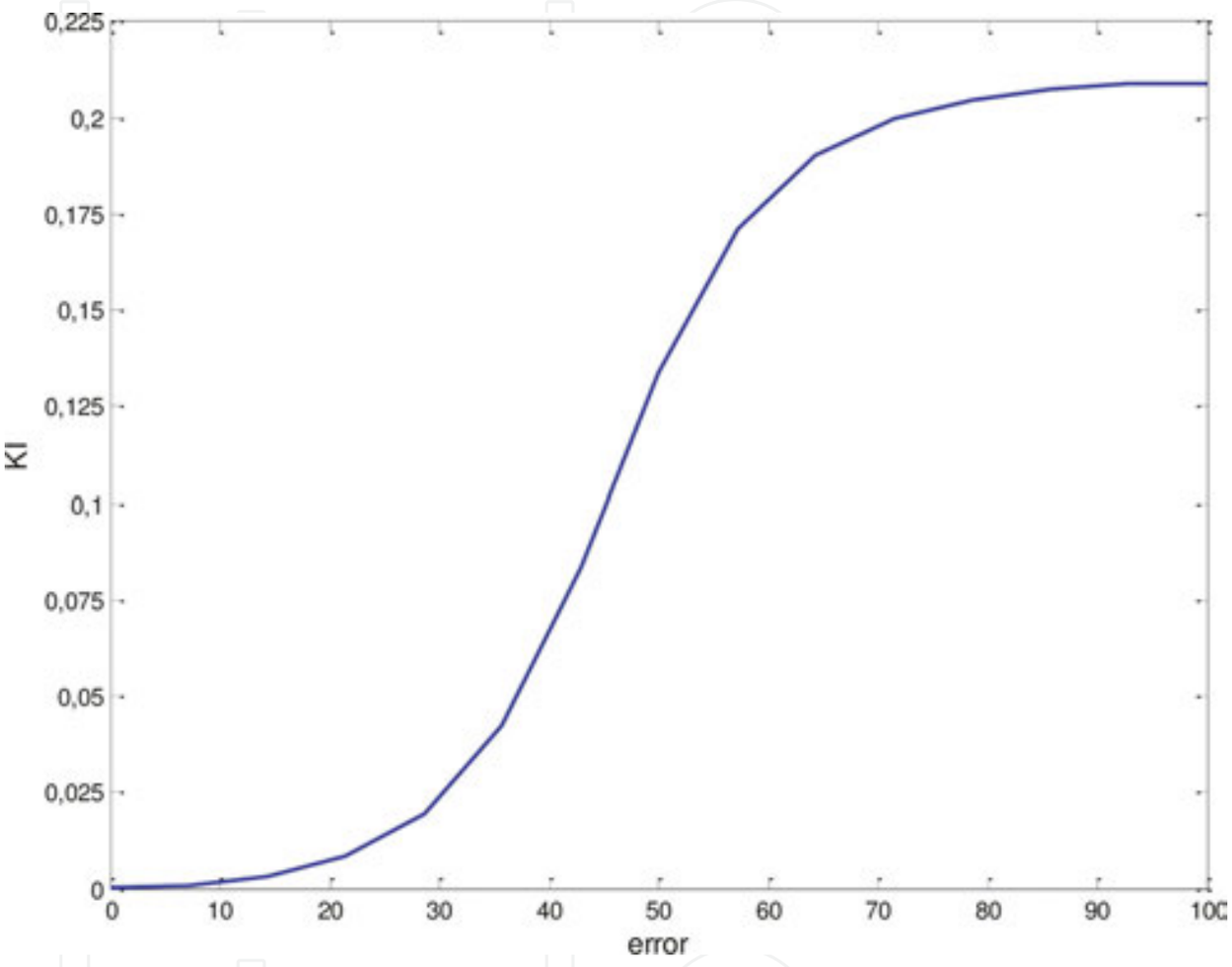

Figure 16. FL surface curve for λ_I.

The feature motions in image plane are shown in **Figure 17.a**, and the velocity signals of the end effector are shown in **Figure 17.b**, respectively. The error signals and the error integral term signals after λ_I product are shown in **Figure 17.c** and **17.d**, respectively. In **Figure 17.b-d**, it is clear that integral term forces fast convergence with high velocities, but this effort is not enough by itself for fast convergence. In (12), it is mentioned that an integral term can be active only when the errors are in predefined bounds if desired. It must be noted that this approach will cause discontinuities and sudden changes in velocity signals of IBVS. The feature motions in **Figure 17.a** show linear characteristics as classical IBVS. The convergence time is 9.41 s and the error cost is 1852 for Case 3.

http://dx.doi.org/10.5772/63046

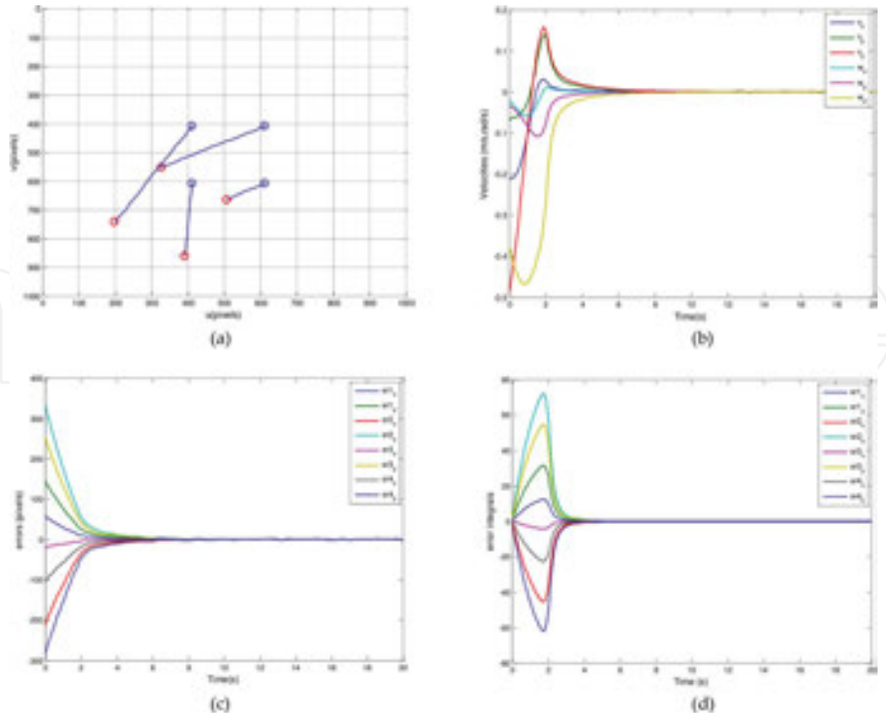

Figure 17. The results for Case 3. (a) Feature motions (b) Velocities of the end effector (c) Feature errors (d) Error integral term.

3.4. Case 4: IBVS with time-varying sliding surface design and constant acceleration

In his study, Bartoszewicz proposed a time-optimal sliding mode approach for robust control of second-order uncertain systems [14]. Time-optimality needs time-varying sliding slopes to adapt the system to initial conditions. These initial conditions for an IBVS system are dominant if they are far from zero. β is chosen as 0.5 for this case and A, B, and C are defined as in (12) and (13), respectively. Time dependent terms in (13) affect velocity limits and by some trials, T in (13) is assigned as 5 s. Inside the *Simulink* block of the sliding surface design for this case is shown in **Figure 18**.

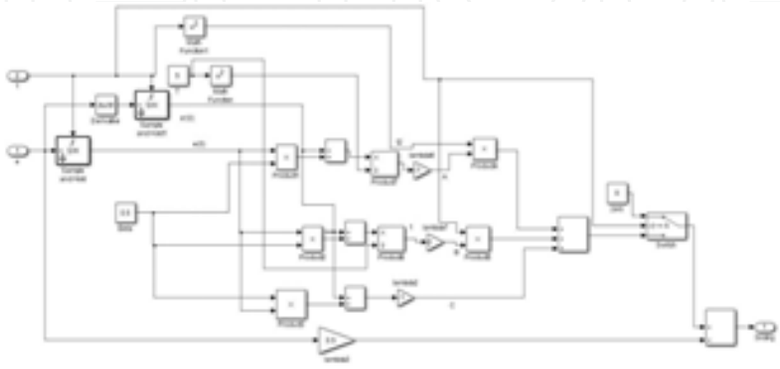

Figure 18. The *Simulink* block of sliding surface for Case 4.

The feature motions in image plane are shown in **Figure 19.a**, the velocity signals of the end effector are shown in **Figure 19.b**, the error signals are shown in **Figure 19.c**, and time dependent terms in (13) are shown in 19.d, respectively.

Figure 19. The results for Case 4. (a) Feature motions (b) Velocities of the end effector (c) Feature errors (d) Time dependent terms in (13).

The feature motion trajectories in the previous cases are quite similar but in this case, the motions are curvilinear as shown in **Figure 19.a**. This is the consequence of time dependent terms shown in **Figure 19.d**. They are the sum of the terms in (13) and they drag the motions throughout target features until the end of time dependence. β adjusts magnitude and signs of these terms in (12) and it has to be chosen wisely. Besides being curvilinear, VS may lose the feature points and stop processing when the features are close to FOV. The convergence time is 11.82 s and the error cost is 2891 for Case 4.

3.5. Case 5: IBVS with nonlinear sliding surface design using tangent hyperbolic function

Tangent hyperbolic function is chosen as nonlinear sliding surface function and the magnitude and slope of this function is also dependent on error and constant c_1 as given in (15). Parameter assignments of (15) are given below:

$$\begin{aligned} w_p(e) &= -0.07|e| + 80 \\ c_1 &= 0.01 \end{aligned} \tag{21}$$

http://dx.doi.org/10.5772/63046

Here, w_p can be defined as a function of time as in (11) but it is chosen as a function of error in this study. Inside the *Simulink* block of the sliding surface design for this case is shown in **Figure 20**.

Figure 20. The *Simulink* block of sliding surface for Case 5.

The feature motions in image plane are shown in **Figure 21.a**, the velocity signals of the end effector are shown in **Figure 21.b**, and the error signals are shown in **Figure 21.c**, respectively. To show the nonlinear sliding surface, error and error derivative trajectories for the first feature in u coordinate is shown in **Figure 21.d**.

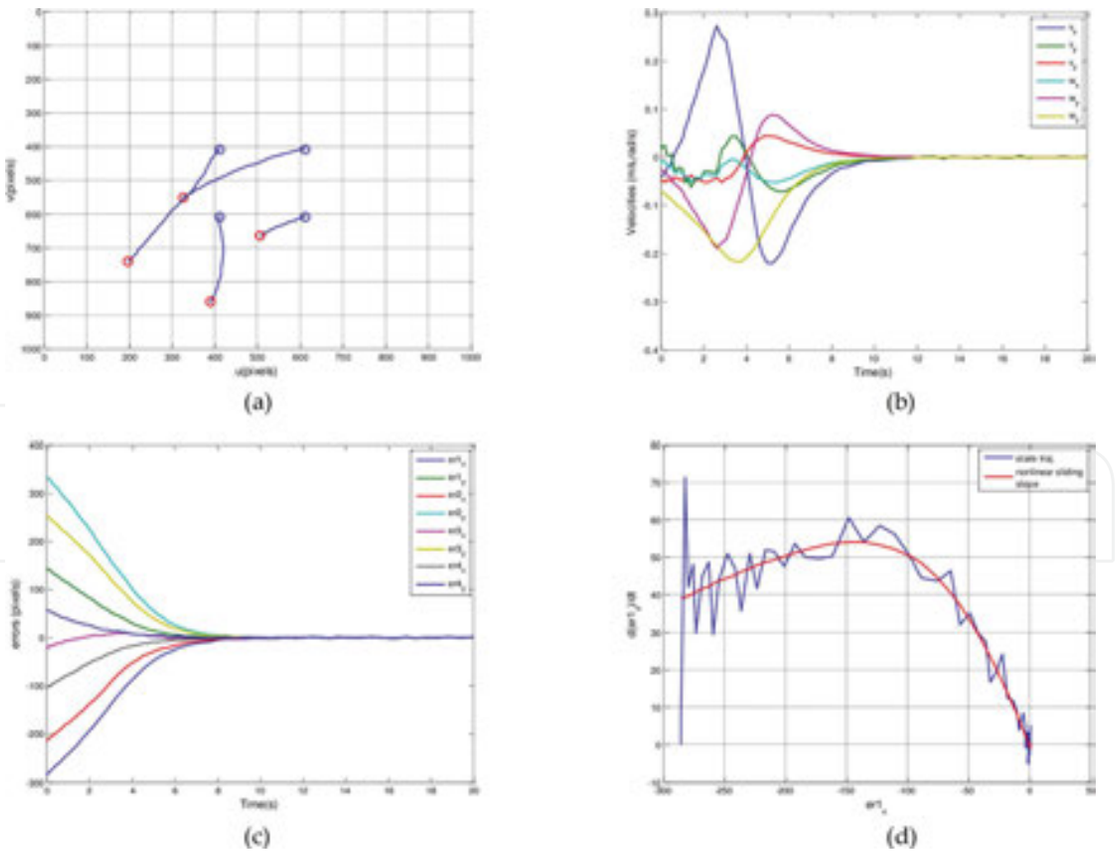

Figure 21. The results for Case 5. (a) Feature motions (b) Velocities of the end effector (c) Feature errors (d) Error vs. error derivative trajectory.

Unlike the first four cases, the initial velocity values are quite small as shown in **Figure 21.b** and that can be a big positive in the realization of an IBVS system as mentioned in [29]. The feature motions are slightly curvilinear as shown in **Figure 21.a**. The convergence time is 10.85 s and the error cost is 4094 for Case 5. Error vs. error derivative trajectory follows tangent hyperbolic function as shown in **Figure 21.d**.

3.6. Case 6: IBVS with nonlinear time-varying sliding surface

Exponential nonlinear time-varying sliding surface in (17) also utilizes initial error and error derivative values. λ is chosen as 0.5 and k exponential time constant is chosen as 1 for a fast transient response. Inside the *Simulink* block of the sliding surface design for this case is shown in **Figure 22**.

The feature motions in image plane are shown in **Figure 23.a**, the velocity signals of the end effector are shown in **Figure 23.b**, and the error signals are shown in **Figure 23.c**, respectively.

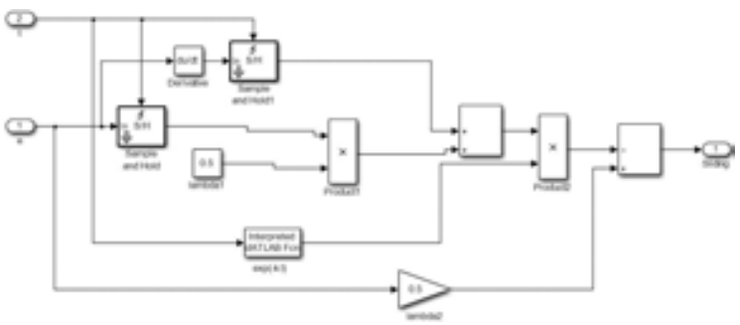

Figure 22. The *Simulink* block of sliding surface for Case 6.

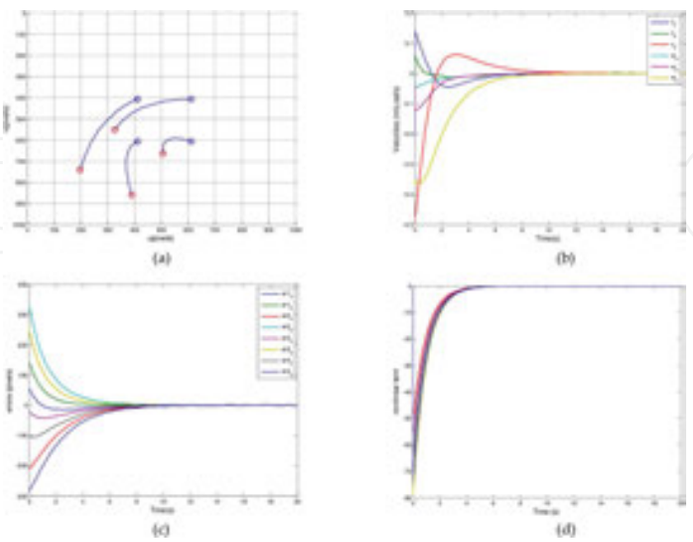

Figure 23. The results for Case 6. (a) Feature motions (b) Velocities of the end effector (c) Feature errors (d) Nonlinear term of each velocity.

http://dx.doi.org/10.5772/63046

Nonlinear term of each velocity in (17) with 5 s transient is shown in Figure 23.d. After this transient, the system behaves like classical IBVS system. The motions are curvilinear as shown in Figure 23.a. The convergence time is 12.12 s and the error cost is 2795 for Case 6.

4. Conclusion and future works

As a new field of control systems, most studies on VS focus on image processing or pattern recognition parts of the applications. The controller part has still gaps. On the other hand, SMC is a convenient control method for linear and nonlinear systems, and it is based on the design of sliding surfaces. In this study, five different sliding surface designs including analytical and intelligent methods are modified and applied to an IBVS manipulator system to adapt the designs and to compare the performance of these designs. The design methods are selected according to their relevance and applicability to this type of visually guided manipulator systems.

Sliding surface design is the first step of SMC and the designer has to consider robustness and transient behaviour of the system. When these designs are considered for an IBVS system, the designer has to deal not only with these issues but also the velocity limits, the convergence time, and motion characteristics. Furthermore, the designer must have experience on sliding surface design to tune the design parameters appropriately. In this study, these hotspots are considered. In the simulations, a random noise is added to each feature point to show the robustness of the designs. According to IBVS metrics mentioned above, a comparison of design methods used in the study is given in **Table 2**.

Design	Convergence time (s)	Error cost	Parameters	Motion characteristic
Classical IBVS with fixed sliding slope	12.89	2842	λ	**Linear**
IBVS with linear varying sliding surface using fuzzy logic	**7.35**	2591	λ_{FL}	**Linear**
IBVS with integral sliding surface using fuzzy logic	9.41	**1852**	λ_P, $\lambda_{I\text{-}FL}$	**Linear**
IBVS with time-varying sliding surface design and constant acceleration	11.82	2891	β, T	Curvilinear
IBVS with nonlinear sliding surface design using tangent hyperbolic function	10.85	4094	w_p (function), c_1	Curvilinear
IBVS with nonlinear time-varying sliding surface	12.12	2795	λ, k	Curvilinear

Table 2. Comparison of sliding surface designs in the study.

All designs in the study considered angular and linear velocity limits which are the most restrictive quantity of a manipulator system, and these limits also form a homogenous basis for comparison. In **Table 2**, it can be seen that IBVS with linear varying sliding surface using

fuzzy logic is the fastest design according to convergence time. It can be concluded that changing sliding slope using FL with error and error derivative inputs results fast response with low velocity profiles. Nonetheless, the only parameter which has to be tuned in this design is not a parameter but an FL unit. As an intelligent method, FL utilizes user or designer experience and the designer has to be careful when using FL in order not to miss a rule. Compared to classical IBVS with fixed sliding slope, all design methods in the study decreases convergence time.

IBVS with integral sliding surface using fuzzy logic has the lowest error cost, but it must be noted that the integral term should be used wisely against wind-up conditions.

Three designs have linear motion characteristics and as mentioned in [1], it is important for keeping FOV. When the other designs with curvilinear characteristics are needed for a VS realization, additional methods should be used to avoid this drawback.

Parameter tuning of each design is another issue binding the theory with user experience. FL units in these designs directly need user experience. In the future studies, intelligent hybrid units which doesn't need user experience like ANFIS will be used.

In the future studies, the main goal will be the realization of these designs in real IBVS manipulator system. Furthermore, not only kinematics of the manipulator, but also the dynamics and inner loop control of the manipulator will be added to the realization.

Author details

Tolga Yüksel

Address all correspondence to: tolga.yuksel@bilecik.edu.tr

Department of Electrical and Electronics Engineering, Bilecik Seyh Edebali University, Bilecik, Turkey

References

[1] Chaumette F, Hutchinson S. Visual servo control. I. Basic approaches. IEEE Robot Autom Mag. 2006;13(4):82–90.

[2] Collewet C, Marchand E, Chaumette F. Visual servoing set free from image processing. Proc - IEEE Int Conf Robot Autom. 2008;81–6.

[3] Kallem V, Swensen JP, Hager GD, Cowan NJ. Kernel-based visual servoing. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. 2007. pp. 1975–80.

[4] Tahri O, Chaumette F. Point-based and region-based image moments for visual servoing of planar objects. IEEE Trans Robot. 2005;21(6):1116–27.

[5] Bourquardez O, Mahony R, Hamel T, Chaumette F. Stability and performance of image based visual servo control using first order spherical image moments. 2006 IEEE/RSJ Int Conf Intell Robot Syst [Internet]. 2006;4304–9. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4059090

[6] Chaumette F, Hutchinson S. Visual servo control. II. Advanced approaches. IEEE Robot Autom Mag [Internet]. 2007;14(1):109–18. Available from: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4141039

[7] Chesi G, Hashimoto K, Prattichizzo D, Vicino A. Keeping features in the field of view in eye-in-hand visual servoing: A switching approach. IEEE Trans Robot. 2004;20(5):908–13.

[8] Allibert G, Courtial E, Chaumette F. Predictive control for constrained image-based visual servoing. IEEE Trans Robot. 2010;26(5):933–9.

[9] Slotine J-J, Li W. Applied Nonlinear Control. Prentice Hall,USA; 1991.

[10] Bandyopadhyay B, Deepak F, Kim K–S. Sliding Mode Control Using Novel Sliding Surfaces. Springer, Berlin Heidelberg; 2009.

[11] Tokat S, Fadali MS, Eray O. A classification and overview of sliding mode controller sliding surface design methods. In: Recent Advances in Sliding Modes: From Control to Intelligent Mechatronics. Springer, Switzerland; 2015.

[12] Stepanenko Y, Cao Y, Su C. Variable Structure Control of Robotic Manipulator With PID Sliding Surfaces. Int J Robust Nonlinear Control. 1998;8:79–90.

[13] Park D-W, Choi S-B. Moving sliding surfaces for high-order variable structure systems. Int J Control. 1999;72(12):960–70.

[14] Bartoszewicz A. Time-varying sliding modes for second-order systems. In: IEE Proceedings - Control Theory and Applications. 1996.

[15] Cerruto E, Consoli A, Kucer P, Testa A. A Fuzzy Logic Quasi-Sliding-Mode Controlled Motor Drive. In: Proceedings of the IEEE International Symposium on Industrial Electronics. 1993. pp. 652–8.

[16] Fridman L, Levant A. Higher Order Sliding Modes. In: Sliding Mode Control in Engineering. 2002. pp. 1–52.

[17] Efe MÖ. Fractional order sliding mode control with reaching law. Turkish J Electr Eng Comput Sci. 2010;18(5):731–47.

[18] Yu S, Yu X, Shirinzadeh B, Man Z. Continuous finite-time control for robotic manipulators with terminal sliding mode. Automatica. 2005;41(11):1957–64.

[19] Kim J-J, Lee J-J, Park K-B, Youn M-J. Design of new time-varying sliding surface for robot manipulator using variable structure controller. Electron Lett. 1993;29(2).

[20] Kaynak O, Erbatur K, Ertugrul M. The fusion of computationally intelligent methodologies and sliding-mode control - A survey. IEEE Trans Ind Electron. 2001;48(1):4–17.

[21] Lee H, Kim E, Kang H-J, Park M. Design of a sliding mode controller with fuzzy sliding surfaces. In: IEE Proceedings - Control Theory and Applications. 1998. pp. 411–8.

[22] Eker I. Sliding mode control with PID sliding surface and experimental application to an electromechanical plant. ISA Trans. 2006;45(1):109–18.

[23] Kowalska O, Kamiński M, Szabat K. Implementation of a sliding-mode controller with an integral function and fuzzy gain value for the electrical drive with an elastic joint. IEEE Trans Ind Electron. 2010;57(4):1309–17.

[24] Mondal S, Mahanta C. Nonlinear sliding surface based second order sliding mode controller for uncertain linear systems. Commun Nonlinear Sci Numer Simul. 2011;16(9):3760–9.

[25] Raman R, Chalanga A, Kamal S, Bandyopadhyay B. Nonlinear sliding surface based adaptive sliding mode control for industrial emulator. In: IEEE International Conference on Industrial Technology (ICIT). 2013. pp. 124–9.

[26] Corke PI. Robotics, Vision & Control: Fundamental Algorithms in Matlab. Springer US; 2011.

[27] Armstrong B, Khatib O, Burdick J. The explicit dynamic model and inertial parameters of the PUMA 560 arm. In: Proceedings 1986 IEEE International Conference on Robotics and Automation. 1986. pp. 510–8.

[28] Sahin T, Zergeroglu E. Adaptive 3D Visual Servo Control of Robot Manipulators via Composite Camera Inputs. Turkish J Electr Eng Comput Sci. 2006;14(2).

[29] Mansard N, Chaumette F. Task sequencing for high-level sensor-based control. IEEE Trans Robot. 2007;23(1):60–72.