symmetry

MDPI

Article

Data-Driven Model Predictive Control for Uncalibrated Visual Servoing

Tianjiao Han, Hongyu Zhu and Dan Yu *

College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China; hantianjiao@nuaa.edu.cn (T.H.); sz2215067@nuaa.edu.cn (H.Z.)
* Correspondence: yudan@nuaa.edu.cn

Abstract: This paper addresses the image-based visual servoing (IBVS) control problem with an uncalibrated camera, unknown dynamics, and constraints. A novel data-driven uncalibrated IBVS (UIBVS) strategy is proposed, incorporated with the Koopman-based model predictive control (KMPC) algorithm and the adaptive robust Kalman filter (ARKF). First, to alleviate the need for calibration of the camera's intrinsic and extrinsic parameters, the ARKF with an adaptive factor is utilized to estimate the image Jacobian matrix online, thereby eliminating the laborious camera calibration procedures and improving robustness against camera disturbances. Then, a data-driven MPC strategy is proposed, wherein the unknown nonlinear dynamic model is learned using the Koopman operator theory, resulting in a linear Koopman prediction model. Only input–output data are used to construct the prediction model, and hence, the proposed approach is robust against model uncertainties. Furthermore, with a symmetric quadratic cost function, the proposed approach solves the quadratic programming problem online, and visibility constraints as well as joint torque constraints are taken into account. As a result, the proposed KMPC scheme can be implemented in real time, and the UIBVS performance degradation which arises from the control torque constraints can be avoided. Simulations and comparisons for a 2-DOF robotic manipulator demonstrate the feasibility of the proposed approach. Simulation results further validate that the computation time of the proposed approach is comparable to the one of kinematic-based methods.

Keywords: robotic control; uncalibrated image-based visual servoing; image Jacobian matrix estimation; data-driven model predictive control

check for updates

Citation: Han, T.; Zhu, H.; Yu, D. Data-Driven Model Predictive Control for Uncalibrated Visual Servoing. *Symmetry* **2024**, *16*, 48. https://doi.org/10.3390/sym16010048

Academic Editors: Lorentz Jäntschi and Michel Planat

Received: 8 November 2023
Revised: 27 December 2023
Accepted: 28 December 2023
Published: 29 December 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Visual servoing control that utilizes visual feedback to guide the movements of the manipulator's end-effector has been widely used in the field of robotics [1]. This approach enhances the manipulator's versatility, expanding its applicability to unstructured environments.

Image-based visual servoing (IBVS) directly controls the manipulator using image feature errors. One class of IBVS approaches realizes visual tracking by directly using global image information for error regulation in the loop [2]. For example, Collewet et al. [3] use photometric information to achieve visual servoing tasks without image feature extraction; however, this method struggles with handling diffuse reflection targets. In addition, other types of global image information, such as histograms [4] or Gaussian mixtures [5], have been utilized.

Another important class of IBVS methods is the point featured-based method. Since the design of point feature-based IBVS methods is simple, they have been widely used in the field of robotic arm visual servoing tasks. Considering the complexity of the calibration process and the impact of camera distortion, uncalibrated image-based visual servoing (UIBVS) methods that do not require precise calibration of the camera's intrinsic and extrinsic parameters have been studied in depth [6–10]. The transformation of feature

points in vision space and the robotic workspace can be represented via the image Jacobian matrix, which is related to depth information, as well as the camera's intrinsic and extrinsic parameters. Therefore, for the UIBVS problem, the control performance highly depends on the accuracy of the image Jacobian matrix estimation. However, the underlying transformation between the pose of the end-effector and the image coordinates of feature points is nonlinear. While the image Jacobian matrix is a linearized representation, it results in modeling uncertainties. Moreover, due to the presence of unknown process and measurement noise, accurately estimating the image Jacobian matrix remains a major challenge in the UIBVS problem.

Currently, most UIBVS methods [11–14] focus on the kinematic-based control, which involves tracking desired image feature points via controlling joint velocities. Despite their advantages in terms of simple controller design and high computation efficiency, these methods require the robotic manipulator to follow commanded speeds in real time, which is often impractical [15]. Furthermore, joint torque constraints are not considered, potentially leading to performance degradation issues.

Dynamic-based control methods have been well studied in robotic control, but they are not widely utilized in UIBVS. One problem resides in the fact that the dynamic model of the robot is nonlinear, which leads to the design of a nonlinear controller with high computation complexity, making real-time visual tracking difficult. In addition, uncertainties arise due to the inaccurate or unknown parameters of the robotic manipulator, such as joint inertia and mass. Designing the controller for systems with unknown dynamics also compromises control precisions. To tackle this problem, data-driven methods [16,17] which do not require the knowledge of the analytical models have been studied recently. However, practical constraints such as joint torque constraints and camera field of view constraints should be taken into account, posing additional challenges to dynamic-based UIBVS control methods.

Motivated by the above-mentioned discussions, in this paper, we propose a novel ARKF-KMPC algorithm for UIBVS control. The ARKF is used to estimate the image Jacobian matrix when both the camera's intrinsic and extrinsic parameters are unknown, which eliminates the laborious camera calibration procedures and enhances the system's robustness against camera disturbances. The Koopman operator is introduced to construct a linear dynamic model of the manipulator, and the MPC controller is used to solve the UIBVS problem with state and control constraints. The main contributions of this paper can be summarized as follows:

- To address the IBVS problem in presence of uncalibrated camera parameters, unknown dynamics, and state and control constraints, we propose an ARKF-KMPC approach in which the UIBVS controller is designed at the dynamic level. Compared to the kinematic-based IBVS methods [18,19], the proposed method is robust against model uncertainties, and performance degradation arising from control torque constraints could be avoided. In contrast to dynamic-based UIBVS methods, the proposed approach alleviates the need for the dynamic model and achieves real-time control performance.
- The proposed ARKF-KMPC approach could effectively handle the camera calibration issue. The ARKF is utilized to estimate the image Jacobian matrix online when both the camera's intrinsic and extrinsic parameters are unknown, thereby eliminating the laborious camera calibration procedures and improving robustness against camera disturbances.
- It is the first time that the data-driven control strategy has been introduced for the UIBVS problem. The unknown nonlinear dynamic model is learned via Koopman operator theory, resulting in a linear Koopman prediction model. Consequently, with a symmetric quadratic cost function, the proposed approach allows for the utilization of quadratic programming (QP) for online optimization, significantly reducing the computation cost, and making the real-time dynamic-based UIBVS control feasible. Simulation results further validate that the computation time of the proposed approach is comparable to the one of kinematic-based methods.

This paper is organized as follows. Section 2 introduces the related work of the paper. Section 3 introduces the mathematical model of the eye-in-hand camera configuration UIBVS system. Section 4 proposes the ARKF-KMPC algorithm for UIBVS. Simulations and comparisons are presented in Section 5, and conclusions are drawn in Section 6.

2. Related Work

UIBVS designs the control law based on the image Jacobian matrix; however, due to the nonlinearity of the robotics and uncertainties in camera parameters, deriving the image Jacobian matrix analytically is nontrivial. Recently, different methods have been proposed to enhance the accuracy and robustness of image Jacobian matrix estimation, such as Kalman filter [20], robust Kalman filter [21], extended Kalman filter [22], particle filter [23], and neural networks [24]. The Kalman filter method was first introduced into the estimation of the image Jacobian matrix by Qian et al. [20]. They constructed a dynamic system wherein the state variables of the system were composed of the elements of the image Jacobian matrix. Subsequently, a Kalman–Bucy filter was employed to online estimate the state variables of the system. The designed filter demonstrated robustness to system noise and external disturbances. Zhong et al. [21] proposed a UIBVS scheme based on the robust Kalman filter, in conjunction with Elman neural network (ENN) learning techniques. The relationship between the vision space and the robotic workspace was learned using an ENN, and then a robust KF was used to improve the ENN learning result. Wang et al. [25] proposed to estimate the total Jacobian matrix using the unscented particle filter, which can make full use of the feature measurements, and hence, can result in more accurate estimations. Gong et al. [23] proposed a geometric particle filter for visual tracking and grasping of moving targets. The introduction of these methods has facilitated research in UIBVS; however, due to the discrepancy between the actual nonlinear transformation and the linearized image Jacobian matrix, the process noise of the image Jacobian matrix estimation model is actually unknown, and hence, how to improve the estimation accuracy without sacrificing real-time performance remains an open problem.

Once the image Jacobian matrix is estimated, different control strategies have been proposed for addressing the UIBVS problem. For example, Chaumette et al. [11] designed the PD controller based on the pseudo-inverse of the image Jacobian matrix. Siradjuddin et al. [12] designed a distributed PD controller of a 7 degrees-of-freedom (DOF) robot manipulator for tracking a moving target. The reinforcement learning algorithm was proposed to adaptively tune the proportional gain in [14]. To address the problem of unknown dynamics, He et al. [26] proposed an eye-in-hand visual servoing control scheme based on the input mapping method, which directly utilizes the past input–output data for designing the feedback control law. The above-mentioned approaches could be used for online control; however, the state and control constraints in the UIBVS system are not taken into account.

Model Predictive Control (MPC) is also a method to solve the constraints in the control system [27–29], which involves the online solution of a finite-horizon open-loop optimization problem at each sampling instant. The first element of the obtained control sequence is then applied, and the process is repeated at the next sampling instant. MPC efficiently handles state and input constraints and could be implemented in real time. Consequently, MPC has been widely used in robotic control, and several MPC-based UIBVS methods have been developed. For example, Qiu et al. [18] proposed an adaptive MPC control scheme wherein the unknown camera intrinsic and extrinsic parameters, unknown depth parameters, as well as external disturbances are estimated via a modified disturbance observer. He et al. [19] proposed a synthetic robust MPC method with input mapping for the UIBVS problem under system constraints. However, these methods are designed only using the kinematics of robotics.

For the dynamic-based MPC methods of the UIBVS system, Qiu et al. [1] proposed an MPC controller based on the identification algorithm and the sliding mode observer. The unknown model parameters are estimated via the parameter identification algorithm, and

the sliding mode observer is designed to estimate the joint velocities of the IBVS system. Wu et al. [30] introduced a two-layer MPC control scheme incorporating an extended state observer to address the problem of parameter uncertainties. Two MPC controllers were designed for both the kinematic level and the dynamic level sequentially, resulting in solving two optimal control problems online. As a result, designing the controller is complicated when tuning design parameters, and the performance of the dynamic controller highly depends on the performance of the kinematic controller. Moreover, both the above dynamic-based MPC methods are based on the nonlinear dynamic model of robotics, which results in solving the nonlinear optimization problem, and hence, is difficult to implement in real time.

Recently, data-driven MPC has been proposed to address the issue of unknown model parameters, wherein input–output data are used to learn the system model. For example, Gaussian Process (GP) [31] and neural networks [32] are employed for offline learning of prediction models, followed by online MPC controller design. However, online updates of GP models require tracking all measurement data, which may be computationally infeasible. Additionally, due to the nonlinearity of the neural network, the controller suffers from real-time implementation.

Koopman operator theory provides a data-driven solution by constructing a linear model of the nonlinear dynamical system, enabling the use of linear control methods like linear quadratic regulators (LQRs) and linear MPC [33]. For instance, Zhu et al. [34] proposed a Koopman MPC (KMPC) approach for trajectory tracking control of an omnidirectional mobile manipulator. Bruder et al. [35] designed MPC controllers for a pneumatic soft robot arm via the Koopman operator and demonstrated that the KMPC controllers outperform the MPC controller based on the linearized model, making accurate linear control of nonlinear systems achievable. However, to the best of the authors' knowledge, few data-driven MPC strategies have been considered for solving the IBVS problems.

3. Preliminaries and Problem Formulation

In this section, the mathematical model of the eye-in-hand camera configuration UIBVS system is introduced, which includes the perspective projection model, image Jacobian matrix, manipulator dynamics model, and visual servoing control model.

3.1. Perspective Projection Control Model

The setup of the eye-in-hand camera configuration is shown in Figure 1. The feature point in the world (base) coordinate system is represented as $M_i = [X_i, Y_i, Z_i]^T (i = 1, \ldots, m)$. According to the perspective projection principle [36], the corresponding image feature points $s_i = [u_i, v_i]^T$ on the image plane are given as

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \frac{1}{Z_i} \begin{bmatrix} \alpha_x & 0 & u_0 & 0 \\ 0 & \alpha_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{T}_e^c \boldsymbol{T}_b^e \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}, \tag{1}$$

where $\begin{bmatrix} \alpha_x & 0 & u_0 & 0 \\ 0 & \alpha_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ is the camera intrinsic matrix. The parameters α_x and α_y represent the focal lengths in two directions in pixels, respectively. $[u_0, v_0]^T$ represents the principal point of the image plane. $\boldsymbol{T}_e^c$ denotes the homogeneous transformation matrix between the end-effector frame and the camera frame. $\boldsymbol{T}_b^e$ denotes the homogeneous transformation matrix of the end-effector with respect to the base frame, which can be calculated via the forward kinematics of the robot manipulator.

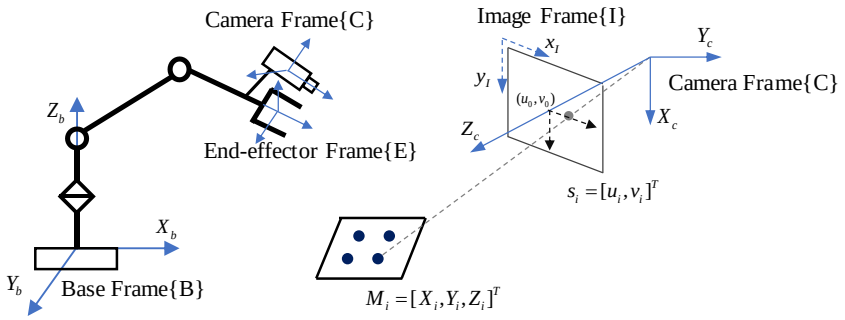

Figure 1. Eye-in-hand camera configuration visual servoing system [19].

3.2. Image Jacobian Matrix

Denote the velocity of the end-effector (camera) as $\dot{\boldsymbol{r}} = (v_x, v_y, v_z, \omega_x, \omega_y, \omega_z)$; then, the velocity of the image feature $[\dot{u}_i, \dot{v}_i]^T$ could be obtained via kinematics of rigid bodies and projection principles as [11]

$$\dot{s}_i = \begin{bmatrix} \dot{u}_i \\ \dot{v}_i \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{\alpha_x}{Z_i} & 0 & \frac{\bar{u}_i}{Z_i} & \frac{\bar{u}_i \bar{v}_i}{\alpha_x} & -\frac{\bar{u}_i^2}{\alpha_x} - \alpha_x & \bar{v}_i \\ 0 & -\frac{\alpha_y}{Z_i} & \frac{\bar{v}_i}{Z_i} & \frac{\bar{v}_i^2}{\alpha_y} + \alpha_y & -\frac{\bar{u}_i \bar{v}_i}{\alpha_y} & -\bar{u}_i \end{bmatrix}}_{\boldsymbol{L}_i} \dot{\boldsymbol{r}}, \tag{2}$$

where $\boldsymbol{L}_i$ is known as the image Jacobian matrix of s_i, and $\bar{u}_i = u_i - u_0$, $\bar{v}_i = v_i - v_0$. The derivation of the image Jacobian matrix is shown in Appendix A.

Denote $\boldsymbol{s} = [s_1, \ldots, s_m]^T$ as the collection of all image feature points; then,

$$\dot{\boldsymbol{s}} = \boldsymbol{L} \cdot \dot{\boldsymbol{r}}, \tag{3}$$

where $\boldsymbol{L} = [\boldsymbol{L}_1^T, \ldots, \boldsymbol{L}_m^T]^T \in \Re^{2m \times 6}$ is the overall image Jacobian matrix.

3.3. Kinematics and Dynamics of Robotic Manipulator

The forward kinematics equation of the robotic manipulator is represented as

$$\dot{\boldsymbol{r}} = \boldsymbol{J}_{\boldsymbol{r}}(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}}, \tag{4}$$

where $\boldsymbol{q} \in \Re^n$ represents the joint angle vector of the manipulator, n is the degree of the robotic manipulator, and $\boldsymbol{J}_{\boldsymbol{r}}(\boldsymbol{q})$ represents the robotic Jacobian matrix.

The dynamic model of the robotic manipulator is given by [37]

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau}, \tag{5}$$

where $\boldsymbol{M}(\boldsymbol{q}) \in \Re^{n \times n}$ represents the symmetric inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \Re^{n \times n}$ represents the centrifugal and Coriolis torque matrix, $\boldsymbol{G}(\boldsymbol{q}) \in \Re^{n \times 1}$ is the gravitational force vector, and $\boldsymbol{\tau} \in \Re^{n \times 1}$ denotes the control input torque vector.

3.4. Image-Based Visual Servoing Control Model

Combining (3) and (4), the relationship between the motion of the feature in image space and the joint velocity in joint space is given as

$$\dot{\boldsymbol{s}} = \boldsymbol{L} \cdot \boldsymbol{J}_{\boldsymbol{r}}(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}}. \tag{6}$$

Let $\boldsymbol{x_1} = \boldsymbol{q}$, $\boldsymbol{x_2} = \dot{\boldsymbol{q}}$; then, from (5) and (6), the overall discrete-time IBVS system model can be written as

$$\boldsymbol{s}(k+1) = \boldsymbol{s}(k) + \boldsymbol{L}(k)\boldsymbol{J_r}(k)\boldsymbol{x_2}(k)\Delta t \tag{7a}$$

$$\boldsymbol{x_1}(k+1) = \boldsymbol{x_1}(k) + \boldsymbol{x_2}(k)\Delta t \tag{7b}$$

$$\boldsymbol{x_2}(k+1) = \boldsymbol{x_2}(k) + \boldsymbol{M}^{-1}(\boldsymbol{\tau}(k) - \boldsymbol{C}\boldsymbol{x_2}(k) - \boldsymbol{G})\Delta t, \tag{7c}$$

where $\boldsymbol{s}(k)$, $\boldsymbol{x_1}(k)$, and $\boldsymbol{x_2}(k)$ represent the image feature points, joint angle, and joint angular velocity at time instant $k\Delta t$, respectively, and Δt is the sampling time.

Therefore, in this paper, the control objective is to design the control input torque $\boldsymbol{\tau}(k)$, such that, without knowing the exact camera intrinsic and extrinsic parameters, the manipulator could track the desired feature points $\boldsymbol{s_d}$.

4. ARKF-KMPC Algorithm for UIBVS

In this section, we propose a novel ARKF-KMPC algorithm for solving the UIBVS problem. The ARKF is introduced to estimate the image Jacobian matrix $\boldsymbol{L}$; then, the Koopman operator is used to approximate the manipulator's dynamic model, and the MPC controller is designed for the linear UIBVS model under state and control constraints. The uncalibrated visual servoing control system framework is shown in Figure 2.

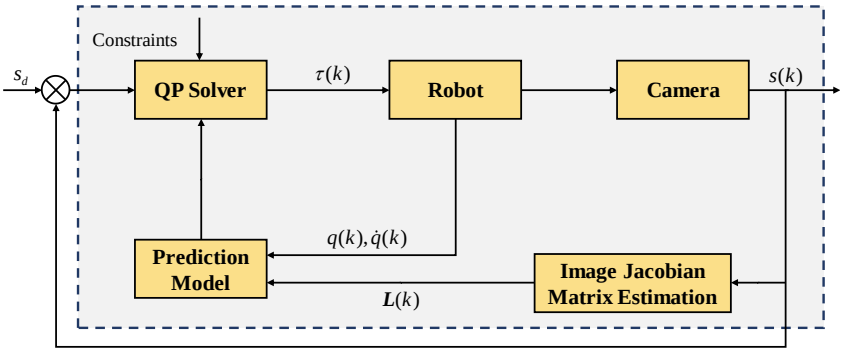

Figure 2. The control framework of the UIBVS system.

4.1. Image Jacobian Matrix Estimation

Since the intrinsic and extrinsic parameters of the camera are unknown or varying, the ARKF algorithm [38] is proposed to estimate the image Jacobian matrix online. Denote $\Phi(k) = [l_{1,1}, \ldots, l_{2m,6}]^T \in \Re^{(2m\times 6)\times 1}$ as an augmented state vector formed by collections of row elements of the image Jacobian matrix $\boldsymbol{L}(k)$. The system observation vector is $Z(k) = \boldsymbol{s}(k) - \boldsymbol{s}(k-1) \in \Re^{2m}$, where $Z(k)$ represents the variation of image features. The image Jacobian matrix could be estimated via the discrete-time system [20]

$$\Phi(k) = \Phi(k-1) + w(k) \tag{8a}$$

$$Z(k) = H(k)\Phi(k) + \nu(k), \tag{8b}$$

where $H(k) \in \Re^{2m\times(2m\times 6)}$ denotes the measurement matrix. $w(k) \in \Re^{(2m\times 6)\times 1}$ and $\nu(k) \in \Re^{2m}$ are the process noise and measurement noise, respectively.

The measurement matrix $H(k)$ could be written as

$$H(k) = \begin{bmatrix} \Delta r(k) & & \\ & \ddots & \\ & & \Delta r(k) \end{bmatrix}_{2m\times(2m\times 6)}. \tag{9}$$

where $\Delta r(k)$ represents the pose change of the camera (end-effector) between time k and $k-1$.

The state and covariance prediction equations are

$$\hat{\Phi}(k|k-1) = \hat{\Phi}(k-1) \tag{10a}$$

$$P(k|k-1) = P(k-1) + Q(k), \tag{10b}$$

where $\hat{\Phi}(k-1)$ and $P(k-1)$ represent the state estimation and error covariance matrix at time $k-1$, respectively. $Q(k)$ is the covariance matrix of the process noise.

Define the error of the state and measurement at time k as

$$\begin{matrix} W(k) = \hat{\Phi}(k) - \hat{\Phi}(k \mid k-1) \\ V(k) = Z(k) - H(k)\hat{\Phi}(k) \end{matrix}. \tag{11}$$

Denote $R(k)$ as the covariance matrix of measurement noise, $P(k|k-1)$ as the error covariance matrix of predicted state vector $\hat{\Phi}(k|k-1)$, and $\alpha(k)$ is an adaptive factor with values $0 < \alpha(k) \leq 1$.

By using the least squares principle,

$$\min(V^T(k)R^{-1}(k)V(k) + \alpha(k)W^T(k)P^{-1}(k|k-1)W(k)), \tag{12}$$

the estimator of the adaptive filter is

$$\hat{\Phi}(k) = \hat{\Phi}(k|k-1) + K(k)[Z(k) - H(k)\hat{\Phi}(k|k-1)], \tag{13}$$

where $K(k)$ is an adaptive gain matrix, and

$$K(k) = \frac{1}{\alpha(k)}P(k|k-1)H^T(k)\left[\frac{1}{\alpha(k)}H(k)P(k|k-1)H^T(k) + R(k)\right]^{-1}. \tag{14}$$

The posterior error covariance matrix of the estimated state vector is

$$P(k) = [I - K(k)H(k)]P(k|k-1)/\alpha(k). \tag{15}$$

Furthermore, the adaptive factor $\alpha(k)$ is constructed based on the prediction residual, which is tuned automatically.

Define the prediction residual vector as

$$\varepsilon(k) = Z(k) - H(k)\hat{\Phi}(k|k-1). \tag{16}$$

The theoretical calculated covariance matrix is

$$C_\varepsilon(k) = H(k)P(k|k-1)H^T(k), \tag{17}$$

and the estimated covariance matrix is

$$\hat{C}_\varepsilon(k) = \frac{1}{N}\sum_{i=1}^{N}\varepsilon(k+1-i)\varepsilon(k+1-i)^T, \tag{18}$$

where N is often chosen as 1 in practice. To verify whether the filter diverges, a statistical measure for discerning the adaptive factor is constructed by comparing the computed matrix $C_\varepsilon(k)$ of $\varepsilon(k)$ with the estimated matrix $\hat{C}_\varepsilon(k)$ as

$$\Delta\varepsilon(k) = \left(\frac{tr(\hat{C}_\varepsilon(k))}{tr(C_\varepsilon(k))}\right)^{1/2}. \tag{19}$$

An adaptive factor function model is constructed as

$$\alpha(k) = \begin{cases} 1 & |\Delta\varepsilon(k)| \le c_0 \\ \frac{c_0}{|\Delta\varepsilon(k)|}\left(\frac{c_0-|\Delta\varepsilon(k)|}{c_1-c_0}\right)^2 & c_0 \le |\Delta\varepsilon(k)| \le c_1 \\ 0 & |\Delta\varepsilon(k)| > c_1 \end{cases}. \tag{20}$$

where c_0 and c_1 are design parameters, and usually, $c_0 = 1.0 \sim 1.5$, $c_1 = 3.0 \sim 8.0$.

The ARKF algorithm for the image Jacobian matrix estimation is summarized in Algorithm 1.

Algorithm 1 ARKF Algorithm.

Input: $\hat{\Phi}(0)$,$P(0)$,$Q(k)$,$R(k)$,c_0,c_1.
Output: $\boldsymbol{L}(k)$.
for $k = 1, 2, \cdots$ do.
Step 1. Predict state $\hat{\Phi}(k|k-1)$ and error covariance $P(k|k-1)$ via (10).
Step 2. Calculate the adaptive factor $\alpha(k)$ via (16)–(20), where $Z(k) = S(k) - S(k-1)$ is the variation of image features, and $H(k)$ is defined in (9).
Step 3. Update the posterior state $\hat{\Phi}(k)$ and error covariance $P(k)$ via (13)–(15).
Step 4. Reorganize $\hat{\Phi}(k)$ for $\boldsymbol{L}(k)$.

Remark 1. *Due to the nonlinearity of the visual servoing system of the robotic manipulator, applying the KF for estimating the image Jacobian matrix needs to satisfy the requirement that the manipulator should not move too fast; otherwise, the linear model (8a) is not valid. Furthermore, the model uncertainties $w(k)$ and $v(k)$ are not known a priori and may not be Gaussian white noise. From (12), it can be seen that the ARKF could adjust the weight of the prediction information, and hence, estimating the image Jacobian matrix via the ARKF could provide more accurate results [38].*

4.2. Koopman Operator for Dynamical System

To tackle the problem where the system's dynamic model is unknown, in this paper, we propose a data-driven modeling approach based on the Koopman theory. We briefly summarize procedures for constructing linear dynamical systems via Koopman operators. For further details, the reader may refer to the work [39,40].

Consider a discrete-time nonlinear dynamical system

$$x(k+1) = f(x(k)), \tag{21}$$

where $x(k) \in \mathcal{X} \subseteq \mathcal{R}^{n_x}$ is the system state, $\mathcal{X}$ is a non-empty compact set, and $f(\cdot) : \mathcal{X} \to \mathcal{X}$ is assumed to be a locally Lipschitz continuous nonlinear function.

Define a real-valued observable function of x as $g(x) : \mathcal{X} \to \mathcal{R}$, and denote $\mathcal{G}$ as the space of observables. Koopman theory maps the nonlinear system (21) to a linear system using an infinite-dimensional linear operator $\mathcal{K} : \mathcal{G} \to \mathcal{G}$, which acts on the observable function, i.e.,

$$\mathcal{K}g(x(k)) = g(f(x(k))) = g(x(k+1)). \tag{22}$$

Therefore, the Koopman operator $\mathcal{K}$ is a linear infinite-dimensional operator which propagates the observable $g(x(k))$ to the next time step. In other words, the Koopman operator evolves nonlinear dynamics linearly without loss of accuracy.

Since the Koopman operator $\mathcal{K}$ is infinite-dimensional, a data-driven algorithm, extended dynamic mode decomposition (EDMD), is used to approximate $\mathcal{K}$ with a finite-dimensional operator $\mathcal{K}_d$ using collected data. A set of observables is used to lift the system from the original state space to a high-dimensional space, and the procedures are summarized as follows.

First, we select N_{lift} observables $g_i, i = 1, \ldots, N_{lift}$, and let

$$g(x) = [g_1(x), g_2(x), \cdots, g_{N_{lift}}(x)]^T. \tag{23}$$

We collect P snapshot pairs $[x(j), x(j+1)], j = 1, \cdots, P$, and construct the lifted data matrices

$$\begin{aligned} X_{1lift} &= \begin{pmatrix} g(x(1)) & g(x(2)) & \cdots & g(x(P)) \end{pmatrix}_{N_{lift} \times P}, \\ X_{2lift} &= \begin{pmatrix} g(x(2)) & g(x(3)) & \cdots & g(x(P+1)) \end{pmatrix}_{N_{lift} \times P}. \end{aligned} \tag{24}$$

Then, the approximate Koopman operator $\mathcal{K}_d$ could be calculated by solving the least-squares problem

$$\min_{\mathcal{K}_d} \| X_{2lift} - \mathcal{K}_d X_{1lift} \|_2^2. \tag{25}$$

where $\| \cdot \|_2$ represents the Euclidean vector norm [41].

Next, we generalize the Koopman theory to the controlled system:

$$x(k+1) = f(x(k), u(k)), \tag{26}$$

where $u(k) \in \mathcal{U} \subseteq \mathcal{R}^{n_u}$ is the control input at time step k. The extended state $\chi(k) = [x(k)^T, u(k)^T]^T \in \mathcal{R}^{n_x + n_u}$ is defined, and the lifted data matrices are further extended as

$$\chi_1 = \begin{pmatrix} X_{1lift} \\ U \end{pmatrix}, \chi_2 = \begin{pmatrix} X_{2lift} \\ U \end{pmatrix}, \tag{27}$$

where $U = [u(1), u(2), \cdots, u(P)]$ such that $x(j+1) = f(x(j), u(j)), j = 1, \cdots, P$. Therefore, the finite-dimensional Koopman operator for the controlled system $\mathcal{K}_{du}$ can be obtained as the solution to the optimization problem

$$\min_{\mathcal{K}_{du}} \| \chi_1 - \mathcal{K}_{du} \chi_2 \|_2^2. \tag{28}$$

Denote the linear Koopman model of (26) as

$$\begin{aligned} z_d(k+1) &= A_d z_d(k) + B_d u(k), \\ \hat{x}(k) &= C_d z_d(k), \end{aligned} \tag{29}$$

where $z_d(k) = g(x(k)) \in \mathcal{R}^{N_{lift}}$ is the lifted state and $\hat{x}(k)$ is the predicted value of the original state $x(k)$. Then, problem (28) is equivalent to

$$\min_{A_d, B_d} \| X_{2lift} - A_d X_{1lift} - B_d U \|_2^2, \tag{30}$$

and the analytical solution is given as

$$[A_d, B_d] = X_{2lift} [X_{1lift}, U]^+, \tag{31}$$

where $(\cdot)^+$ denotes the pseudoinverse of matrix $(\cdot)$. C_d is obtained by solving the least-squares problem

$$\min_{C_d} \| X - C_d X_{1lift} \|_2^2, \tag{32}$$

where $X = [x(1), x(2), \cdots, x(P)]$, and the solution is given as

$$C_d = X X_{1lift}^+. \tag{33}$$

Furthermore, we choose observable functions

$$g(x) = [x^T, g_{n_x+1}(x), \cdots, g_{N_{lift}}(x)]^T, \tag{34}$$

i.e., the observable functions contain the original state; then,

$$C_d = [I_{n_x \times n_x}, 0]. \tag{35}$$

The Koopman linear system identification algorithm is summarized in Algorithm 2.

Algorithm 2 Koopman Linear System Identification.

Input: N_{lift}.
Output: A_d, B_d,C_d.
Step 1. Collect P snapshot pairs $[x(j), x(j+1)]$ and $u(j)$ for $j = 1, \cdots, P$, where $x(j) = [x_1(j), x_2(j)]^T$ and $u(j) = \tau(j)$.
Step 2. Choose N_{lift} observable functions $g_i, i = 1, \cdots, N_{lift}$ and construct the lifted data matrices via (25) and (28).
Step 3. Identify the linear Koopman model (31) via (32) and (34).

Remark 2. *To train the Koopman model, we collect P input-state datasets* $[x(j), x(j+1), u(j)]_{j=1}^{P}$, *satisfying (26), and the data need not to be collected at consecutive time steps. The dimension of the lifted state space* N_{lift} *is a design parameter, which should be selected according to the precision requirements. The observable functions* g_i *can be chosen as some basis functions, such as Gaussian kernel functions, polyharmonic splines, and thin plate splines, or neural networks.*

Remark 3. *A standard algorithm [39] is used to realize the Koopman model (29), and modifications of the algorithm are beyond the scope of this research. Recently, different algorithms have been proposed to improve the estimation accuracy of the model (29). For example, deep neural networks have been used to learn the observable functions [42], and instructions to construct a stable Koopman model have been studied in [43]. Stability analysis has been discussed in [43,44].*

4.3. KMPC for UIBVS

Consider the nonlinear dynamic model of the manipulator (7b) and (7c), and let $\boldsymbol{x}(k) = [\boldsymbol{x_1}^T(k), \boldsymbol{x_2}^T(k)]^T \in \Re^{2n}$. We choose observable functions to contain the original state $\boldsymbol{x}(k)$ as (34); then, the linear Koopman model (29) could be constructed, where $z_d(k) = [\boldsymbol{x_1}^T(k), \boldsymbol{x_2}^T(k), g_R^T(x(k))]^T$, and $g_R^T(x(k)) \in \Re^{N_{lift}-2n}$ represents the rest of the lifted states. Therefore, the overall linear UIBVS prediction model is

$$\begin{aligned} \tilde{z}_d(k+1) &= \tilde{A}_d \tilde{z}_d(k) + \tilde{B}_d \tau(k), \\ \tilde{x}_d(k) &= \tilde{C}_d \tilde{z}_d(k), \end{aligned} \tag{36}$$

where

$$\begin{aligned} \tilde{A}_d &= \begin{bmatrix} I_{2m\times 2m} & 0_{2m\times n} & \boldsymbol{L}\boldsymbol{J_r}\Delta t & 0_{2m\times(N_{lift}-2n)} \\ 0_{N_{lift}\times 2m} & & A_{d_{N_{lift}\times N_{lift}}} & \end{bmatrix}, \\ \tilde{B}_d &= \begin{bmatrix} 0 \\ B_d \end{bmatrix}, \\ \tilde{C}_d &= \begin{bmatrix} I_{2m\times 2m} & 0 \\ 0 & C_{d_{2n\times N_{lift}}} \end{bmatrix}, \end{aligned} \tag{37}$$

and $\tilde{z}_d(k) = \begin{bmatrix} \boldsymbol{s}(k) \\ z_d(k) \end{bmatrix}$ is the extended state vector, $\tilde{x}_d(k) = \begin{bmatrix} \boldsymbol{s}(k) \\ \hat{x}(k) \end{bmatrix}$ is the extended predicted value of states.

In the UIBVS, the image feature should remain within the image plane, so that the visibility constraints are defined as

$$\boldsymbol{s_{min}} \leq \boldsymbol{s}(k) \leq \boldsymbol{s_{max}}, \tag{38}$$

where $\boldsymbol{s_{min}}$ and $\boldsymbol{s_{max}}$ are the image boundaries, respectively.

The control input constraints are defined as

$$\boldsymbol{\tau_{min}} \leq \boldsymbol{\tau(k)} \leq \boldsymbol{\tau_{max}} \tag{39a}$$

$$\Delta\boldsymbol{\tau_{min}} \leq \Delta\boldsymbol{\tau(k)} \leq \Delta\boldsymbol{\tau_{max}}, \tag{39b}$$

where $\boldsymbol{\tau_{min}}$ and $\boldsymbol{\tau_{max}}$ are the minimum and maximum control input torques, respectively. $\Delta\boldsymbol{\tau_{min}}$ and $\Delta\boldsymbol{\tau_{max}}$ are the minimum and maximum control input increments, respectively.

Define the quadratic cost function at sampling time k as

$$\begin{aligned} J(k) = &\sum_{i=1}^{N_p-1} \|\boldsymbol{s_d} - \boldsymbol{s}(k+i|k)\|_{F^*}^2 + \left\|\boldsymbol{s_d} - \boldsymbol{s}(k+N_p|k)\right\|_{R^*}^2 \\ &+ \sum_{i=1}^{N_p} \|\boldsymbol{\tau}(k+i-1|k)\|_{G^*}^2, \end{aligned} \tag{40}$$

where N_p is the prediction horizon, F^*, G^*, and R^* are the weighting matrices. $\boldsymbol{s}(k+i|k)$ denotes the predicted image feature points at time $k+i$ via prediction model (36) given the current state $\boldsymbol{s}(k)$, $\boldsymbol{\tau}(k+i-1|k)$ denotes the input at the time instant $k+i-1$.

The object of the UIBVS is to track the desired feature points $\boldsymbol{s_d}$, and the optimization problem is formulated as follows:

$$\begin{aligned} &\min_{\{\boldsymbol{\tau}(k+i-1|k)\}_{i=1}^{N_p}} J(k), \\ &\text{s.t.} \quad (36), (38), (39) \end{aligned} \tag{41}$$

At each sampling time, the optimal input sequence $\{\boldsymbol{\tau}(k+i-1|k)\}_{i=1}^{N_p}$ is calculated by solving the open-loop finite-horizon optimal control problem. The first element of the sequence is considered as the optimal control input $\boldsymbol{\tau}^*(k)$, and is applied to the robot. The process is repeated at the next sampling moment. Since $J(k)$ is a symmetric quadratic cost function, and the prediction model (36) is linear, the quadratic programming (QP) algorithm is used to solve the optimal control problem above.

Remark 4. *Compared to conventional visual servoing control methods [7,12,45], MPC is suitable for addressing situations with state and control constraints. Furthermore, in contrast to approaches in [18,30], the proposed method directly controls the joint torques of robotics, which can improve the control accuracy and robustness to external disturbances, enabling the control system to better adapt to complex working environments. Additionally, the symmetric cost function and the Koopman MPC strategy used in this paper result in a linear quadratic optimization problem, which could be solved via QP. This leads to a notable increase in computation speed, ensuring that real-time requirements are met and substantially enhancing the practical applicability of the proposed method.*

The proposed ARKF-KMPC algorithm for UIBVS is summarized in Algorithm 3.

Algorithm 3 ARKF-KMPC Algorithm.

Input: Prediction horizon N_p, cost matrices F^*, G^*, R^*.
Output: Control input τ^*.

Step 1. Koopman Linear System Identification. Identify the Koopman linear model A_d, B_d, C_d via Algorithm 2.
Step 2. for $k = 0, 1, 2, \ldots$ **do.**
a. **Image Jacobian Matrix Estimation.** Estimate $\boldsymbol{L}(k)$ via Algorithm 1, and update $\tilde{A}_d$ in (37).
b. **KMPC optimization.** Solve (41) via QP to find the optimal input $\tau^*(k)$.
c. Apply $\tau^*(k)$ to the UIBVS system.

5. Simulation Results

We test the proposed ARKF-KMPC method on a 2-DOF robotic manipulator. The parameters of the manipulator are shown in Table 1. The inertia matrix $\boldsymbol{M}(\boldsymbol{q})$, the centripetal and Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and the gravitational torque vector $\boldsymbol{G}(\boldsymbol{q})$ are defined in Appendix B [30].

Table 1. Parameters of the manipulator.

Notation	Parameters	Value	Unit
Length of link 1	l_1	0.45	m
Center length of link 1	l_{c1}	0.091	m
Center length of link 2	l_{c2}	0.048	m
Mass of link 1	m_1	23.902	kg
Mass of link 2	m_2	3.880	kg
Inertia of link 1	H_1	1.266	kg m^2
Inertia of link 2	H_2	0.093	kg m^2
Gravity acceleration	g	9.81	m/s^2

Simulations are carried out with unknown dynamics, and the camera intrinsic parameters are unknown. The actual intrinsic parameters of the camera are set as follows: the focal length $f = 0.008$ m, the coordinates of the principal points $(u_0, v_0) = (512, 512)$ pixels. The image has a 1024×1024 pixels resolution. The sampling time is 0.01s. The simulations are conducted in MATLAB with a 2.6 GHz Intel Core i5.

In general, root mean squared deviation (RMSD) is defined as

$$RMSD = \sqrt{\frac{1}{N_{sim}} \sum_{k=1}^{N_{sim}} \|\Psi(k) - \bar{\Psi}(k)\|_2^2}, \tag{42}$$

where $\Psi(k) \in \Re^{n_0 \times 1}$ is any n_0-dimensional actual state vector at time step k, and $\hat{\Psi}(k) \in \Re^{n_0 \times 1}$ represents the corresponding predicted state vector.

5.1. Verification of the Model Accuracy

First, we test the prediction accuracy of the Koopman model.

Training of the Koopman model. Data are collected over 1000 trajectories, with 100 snapshots in each trajectory. The control input for each trajectory is generated from a two-dimensional Gaussian distribution with zero mean and covariance of $2I_{2\times2}$. The initial condition for each trajectory is randomly generated with a uniform distribution over $[-1, 1]$. The observable functions are chosen to be the state itself, and the thin plate spline radial basis functions, where the center points are uniformly distributed in a given interval $[-1, 1]$. The total number of observable states is N_{lift}.

In Table 2, we compare the prediction performance of different Koopman models during a short duration of 0.15 s, a medium duration of 0.5 s, and a long duration of 2 s. For each model, a Monte Carlo simulation with 1000 randomly generated initial conditions is conducted, and the RMSDs over the entire prediction horizon are shown in Table 2 as a function of N_{lift}.

It should be noted that prediction errors do not necessarily decrease as N_{lift} increases, and the prediction performance for the short, medium, and long durations are different. For example, the best prediction performance can be achieved when $N_{lift} = 69$ for the short duration but it performs the worst for the long duration. Therefore, the design parameter N_{lift} should be tuned by trial-and-error.

Table 2. Comparisons of modeling errors.

N_{lift} \ $RMSD$	$T_{max}(s)$		
	0.15 s	**0.5 s**	**2.0 s**
65	0.1901	0.4390	1.6361
66	0.1884	0.3753	0.9543
69	0.1862	0.3754	2.5227
77	0.1927	0.5617	1.8364
80	0.1948	0.5410	2.5043
86	0.1862	0.3814	2.0103
102	0.1862	0.4226	1.9516

Testing of the Koopman model. In Figure 3, we compare the prediction results between the original nonlinear dynamic model (7b) and (7c), and the proposed Koopman model with $N_{lift} = 66$ within a duration of 0.15s. It can be seen that predictions using the Koopman model are rather accurate.

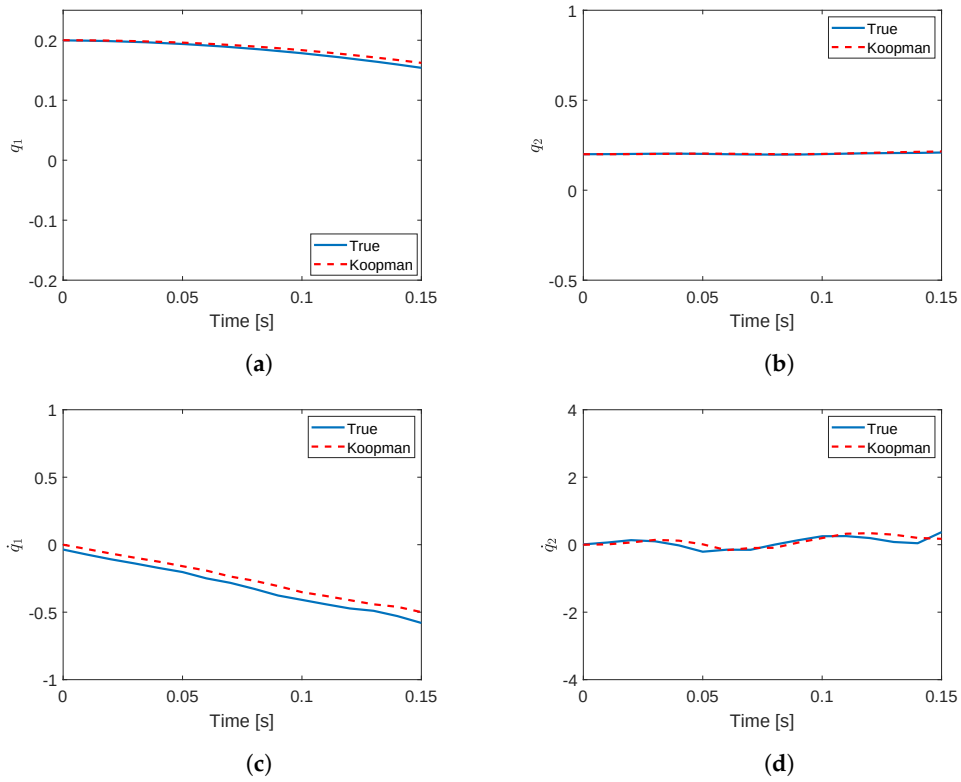

Figure 3. Prediction results for a 2-DOF robotic manipulator. (**a**) Joint angle q_1. (**b**) Joint angle q_2. (**c**) Joint angular velocity $\dot{q}_1$. (**d**) Joint angular velocity $\dot{q}_2$.

5.2. Verification the Control Performance of the Model

Next, we test the control performance of the proposed ARKF-KMPC method. The visibility constraints are set as

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} pixels \leq \boldsymbol{s} \leq \begin{bmatrix} 1024 \\ 1024 \end{bmatrix} pixels. \tag{43}$$

The control input constraints are set as

$$\begin{bmatrix} -0.5 \\ -0.5 \end{bmatrix} Nm \leq \Delta\boldsymbol{\tau} \leq \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} Nm \tag{44a}$$

$$\begin{bmatrix} -10 \\ -10 \end{bmatrix} Nm \leq \boldsymbol{\tau} \leq \begin{bmatrix} 10 \\ 10 \end{bmatrix} Nm. \tag{44b}$$

The control objective is to map four feature points from their initial positions to desired positions on the image plane, with an error tolerance of four pixels, i.e.,

$$\|e(k)\|_2 = \|\boldsymbol{s}(k) - \boldsymbol{s_d}\|_2 \leq 4. \tag{45}$$

The initial positions of the feature points on the image plane are $(529.5, 97.8)^T$ pixels, $(729.5, 297.8)^T$ pixels, $(729.5, 97.8)^T$ pixels, and $(529.5, 297.8)^T$ pixels, respectively. The desired positions of the feature points on the image plane are $(454.8, 377.7)^T$ pixels, $(654.8, 577.7)^T$ pixels, $(654.8, 377.7)^T$ pixels, and $(454.8, 577.7)^T$ pixels, respectively. For image Jacobian matrix estimation, the weighting matrices are chosen as $P = 10I_{48\times48}$, $Q = 0.1I_{8\times8}$, and $R = 0.1I_{8\times8}$, and the design parameters are $c_0 = 1.0$ and $c_1 = 3.0$. For the MPC controller, the weighting matrices are chosen as $F^* = I_{8\times8}$, $G^* = 100I_{8\times8}$, and $R^* = 0.1I_{2\times2}$. To balance the computation time and control efficiency, the prediction horizon is chosen as $N_p = 0.15$ s. The dimension of the Koopman model is chosen as $N_{lift} = 66$.

Figure 4 shows simulation results for a 2-DOF robotic manipulator. The error curve $\|e(k)\|_2$ between the four actual feature points and the desired ones is shown in Figure 4a, where the smallest error is 3.7793 pixels. In Figure 4b, the 2-D trajectory of the feature points on the image plane is presented. It can be seen that the robotic manipulator can reach the desired position successfully. The control input torques of the joints are shown in Figure 4c, from which we can observe that the joint torques and the increment of joint torques are always within the constraints. Joint angles of the 2-DOF manipulator are shown in Figure 4d, which changes smoothly. The average optimization time at each time step is 0.0009 s, comfortably meeting real-time requirements.

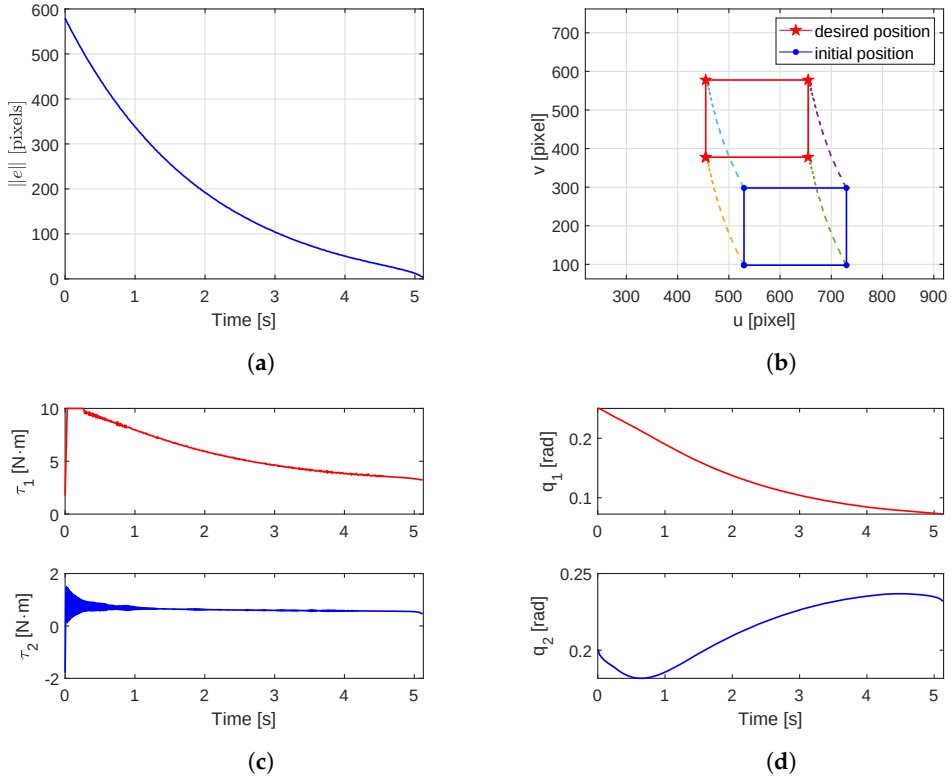

Figure 4. Simulation results for a 2-DOF robotic manipulator. (**a**) Image errors. (**b**) Image trajectory. (**c**) Control input torque. (**d**) Joint angles.

Furthermore, we conducted 100 Monte Carlo simulations for models with different values of N_{lift}. Table 3 presents the average final control errors for different models. The results further corroborate the effectiveness of the proposed control algorithm.

Table 3. Control errors for different models.

Model(N_{lift})	65	66	69	77	80	86	102
Error(pixels)	3.7723	3.7641	3.8262	3.2758	3.1437	3.8592	3.7712

We also compare the proposed method with the method using the original nonlinear dynamics model. Although the image feature points using the nonlinear dynamic model converge to the desired ones with a smaller error of 2.72 pixels, the average computation time is 5.4201 s, which results from solving the nonlinear optimization problem using fmincon, and hence, is not suitable in real applications.

Furthermore, to validate the performance of the proposed ARKF algorithm for image Jacobian matrix estimation, we compare the RMSD between the estimated image Jacobian matrix and the actual image Jacobian matrix using KF and ARKF. We perform 100 Monte Carlo simulations; in each simulation, the control inputs are randomly generated from a two-dimensional Gaussian distribution with zero mean and a covariance of $2I_{2\times2}$, over a simulation time of 6 s. The average RMSD of 100 Monte Carlo simulations using the KF algorithm is 1.3716 $\times 10^3$ and 1.1995 $\times 10^3$ using the ARKF. The RMSDs in 100 Monte Carlo simulations for both algorithms are presented in Figure 5.

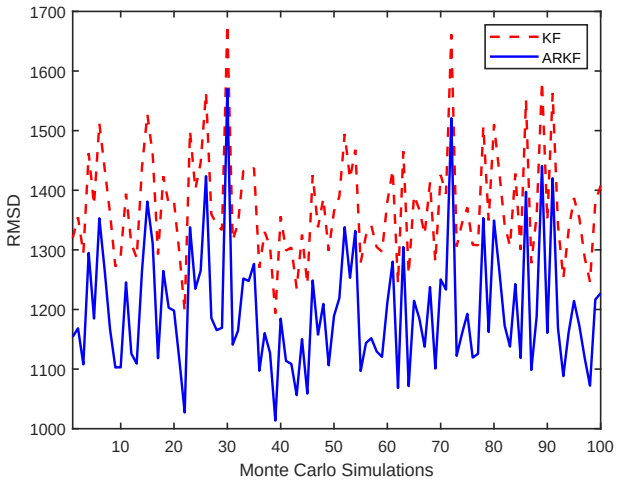

Figure 5. Comparison of RMSDs of the estimated image Jacobian matrix and the actual image Jacobian matrix.

It can be seen that, compared to the KF method, applying the ARKF for estimating the image Jacobian matrix effectively reduces the estimation error, enabling the system to perform visual servoing tasks accurately.

5.3. Comparisons with Related Work

To further demonstrate the superiority of the proposed algorithm, a comparative analysis is conducted against three related methods, as detailed in Table 4. For all methods, the kinematic-based MPC is solved using QP, and design parameters are all tuned for best performance.

Table 4. Comparison with existing methods.

Methods	Kinematics	Dynamics	Error (pixels)	Time (s)
(a)	MPC	×	7.3696	0.0005
(b)	MPC	PID	8.9505	0.0014
(c)	MPC	MPC	18.3579	1.5778
Ours	×	KMPC	3.7793	0.0009

Method (a) designs the MPC controller only considering the kinematics of the manipulator, resulting in the fastest convergence rate. The average computation time for this method is about 0.0005 s per time step, which is the lowest among four algorithms; however, the final image error is 7.3696 pixels, which cannot satisfy the control performance requirement.

Both Methods (b) and (c) design two controllers for kinematics and dynamics separately. The joint angular velocities are optimized via the kinematic-based MPC controller, and the PD controller and MPC controller are designed at the dynamic level to track the desired joint angular velocities, respectively. It can be seen that, without sacrificing too much computation time, the position control using the PD controller is feasible, and the final image error of Method (b) is comparable to that of Method (a); however, the joint torque constraints are not considered when using the PD controller. On the other hand, Method (c) needs to solve a nonlinear optimization problem at each time step, and the average computation time is around 1.5778 s, which could not be implemented in real time. Moreover, the control performance of both Methods (b) and (c) highly depends on the results from the kinematic-based MPC controller, and the selection of design parameters also poses challenges in this problem.

Therefore, from Table 4, it can be seen that the proposed ARKF-KMPC method, which designs the dynamic-based MPC controller directly, could achieve the best performance. With a linear Koopman prediction model and a symmetric quadratic cost function, a linear optimization problem is solved at each time step, the computation time of the proposed approach is comparable to the one of the kinematic-based MPC controller and is significantly reduced when compared to Method (c). Moreover, the proposed approach is a data-driven control approach which does not need the actual dynamic model, making it robust to model uncertainties.

6. Conclusions

In this paper, we propose an ARKF-KMPC scheme for the data-driven control of constrained UIBVS systems. First, when the intrinsic and extrinsic parameters of the camera are unknown, the ARKF with an adaptive factor is utilized to estimate the image Jacobian matrix online. Then, the KMPC strategy is proposed to solve the UIBVS problem at the dynamic level. To address the issue of unknown dynamics, a linear Koopman prediction model is constructed via input–output data offline, resulting in a linear optimal control problem which is solved via QP online. The proposed UIBVS strategy is robust to imprecise camera calibration as well as unknown dynamics, with state and control constraints taken into account. Numerical simulations demonstrate that the proposed approach outperforms the kinematic-based control approaches, and the computation time of the proposed approach is comparable to the one of the kinematic-based control approach and has been significantly reduced when compared to the dynamic-based control approaches. The existing problem when estimating the image Jacobian matrix via the ARKF algorithm with a linear discrete-time system model is that the robotic manipulator should not move too fast. Therefore, the application of the proposed approach may be limited and the construction of a more realistic model for estimating the image Jacobian matrix could be further studied in future work. Also, the proposed approach is validated using the

simulation, and future work will include testing the proposed method using data collected from actual experiments.

Author Contributions: Conceptualization, T.H. and D.Y.; methodology, T.H. and H.Z.; software, T.H.; validation, T.H. and D.Y.; writing—original draft preparation, T.H.; writing—review and editing, D.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by Nanjing University of Aeronautics and Astronautics (Grant number xcxjh20221509).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Image Jacobian Matrix Definition

Denote the coordinates of the feature points M_i in the world coordinate system with respect to the camera is $M_i = [X_i, Y_i, Z_i]^T$; then, the projection of the feature points on the image plane is

$$\begin{aligned} x_{i,I} &= \frac{fX_i}{Z_i} \\ y_{i,I} &= \frac{fY_i}{Z_i}. \end{aligned} \tag{A1}$$

where f is the focal length.

Time derivatives of Equation (A1) are

$$\begin{aligned} \dot{x}_{i,I} &= f\frac{Z_i\dot{X}_i - X_i\dot{Z}_i}{Z_i^2} \\ \dot{y}_{i,I} &= f\frac{Z_i\dot{Y}_i - Y_i\dot{Z}_i}{Z_i^2}, \end{aligned} \tag{A2}$$

which could be written in a matrix form as

$$\begin{bmatrix} \dot{x}_{i,I} \\ \dot{y}_{i,I} \end{bmatrix} = \begin{bmatrix} \frac{f}{Z_i} & 0 & -\frac{fX_i}{Z_i^2} \\ 0 & \frac{f}{Z_i} & -\frac{fY_i}{Z_i^2} \end{bmatrix} \begin{bmatrix} \dot{X}_i \\ \dot{Y}_i \\ \dot{Z}_i \end{bmatrix}. \tag{A3}$$

The speed of the feature points with respect to the camera is

$$\dot{M}_i = -\omega \times M_i - v. \tag{A4}$$

Expanding (A4) leads to

$$\begin{aligned} \dot{X}_i &= Y_i\omega_z - Z_i\omega_y - v_x \\ \dot{Y}_i &= Z_i\omega_x - Z_i\omega_z - v_y \\ \dot{Z}_i &= X_i\omega_y - Z_i\omega_x - v_z. \end{aligned} \tag{A5}$$

Rewrite (A5) into a matrix form as

$$\begin{bmatrix} \dot{X}_i \\ \dot{Y}_i \\ \dot{Z}_i \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & -Z_i & Y_i \\ 0 & -1 & 0 & Z_i & 0 & -X_i \\ 0 & 0 & -1 & -Y_i & X_i & 0 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ w_x \\ w_y \\ w_z \end{bmatrix}. \tag{A6}$$

The relationship between the known image coordinate system and the pixel coordinate system is

$$\begin{aligned} u_i &= \frac{f}{\rho_x} \cdot x_{i,I} + u_0 \\ v_i &= \frac{f}{\rho_y} \cdot y_{i,I} + v_0, \end{aligned} \tag{A7}$$

where ρ_x and ρ_y are the width and height of each pixel, respectively.

According to (A1), (A3), (A6), and (A7), we can obtain

$$\dot{s}_i = \begin{bmatrix} \dot{u}_i \\ \dot{v}_i \end{bmatrix} = \underbrace{\begin{bmatrix} -\frac{\alpha_x}{Z_i} & 0 & \frac{\bar{u}_i}{Z_i} & \frac{\bar{u}_i \bar{v}_i}{\alpha_x} & -\frac{\bar{u}_i^2}{\alpha_x} - \alpha_x & \bar{v}_i \\ 0 & -\frac{\alpha_y}{Z_i} & \frac{\bar{v}_i}{Z_i} & \frac{\bar{v}_i^2}{\alpha_y} + \alpha_y & -\frac{\bar{u}_i \bar{v}_i}{\alpha_y} & -\bar{u}_i \end{bmatrix}}_{\boldsymbol{L}_i} \dot{\boldsymbol{r}}, \tag{A8}$$

where $\alpha_x = \frac{f}{\rho_x}$ and $\alpha_y = \frac{f}{\rho_y}$, $\boldsymbol{L}_i$ is known as the image Jacobian matrix of s_i, and $\bar{u}_i = u_i - u_0$, $\bar{v}_i = v_i - v_0$.

Appendix B. 2-DOF Robotic Manipulator Model

The entries $\boldsymbol{M}_{ij}(q)(i,j = 1,2)$ in the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ are given by

$$\begin{aligned} \boldsymbol{M}_{11}(\boldsymbol{q}) &= H_1 + m_1 l_{c1}^2 + m_2 (l_1^2 + l_{c2}^2 + 2 l_1 l_{c2} cos(q_2)) \\ \boldsymbol{M}_{12}(\boldsymbol{q}) &= H_2 + m_2 \left(l_{c2}^2 + l_1 l_{c2} cos(q_2) \right) \\ \boldsymbol{M}_{21}(\boldsymbol{q}) &= H_2 + m_2 \left(l_{c2}^2 + l_1 l_{c2} cos(q_2) \right) \\ \boldsymbol{M}_{22}(\boldsymbol{q}) &= H_2 + m_2 l_{c2}^2. \end{aligned} \tag{A9}$$

The elements in the centripetal and Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ are given by

$$\begin{aligned} \boldsymbol{C}_{11}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= 2 m_2 l_1 l_{c2} sin(q_2) \dot{q_2} \\ \boldsymbol{C}_{12}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= -m_2 l_1 l_{c2} sin(q_2) \dot{q_2} \\ \boldsymbol{C}_{21}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= m_2 l_{c2} sin(q_2) \dot{q_1} \\ \boldsymbol{C}_{22}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= 0 \end{aligned}, \tag{A10}$$

and the elements in the gravitational torque vector $\boldsymbol{G}(\boldsymbol{q})$ are

$$\begin{aligned} \boldsymbol{G}_{11}(\boldsymbol{q}) &= g(m_1 l_{c1} + m_2 l_1) sin(q_1) + g m_2 l_{c2} sin(q_1 + q_2) \\ \boldsymbol{G}_{21}(\boldsymbol{q}) &= g m_2 l_{c2} sin(q_1 + q_2) \end{aligned}, \tag{A11}$$

where q_i denotes ith joint of the manipulator.

References

1. Qiu, Z.; Hu, S.; Liang, X. Model Predictive Control for Uncalibrated and Constrained Image-Based Visual Servoing without Joint Velocity Measurements. *IEEE Access* **2019**, *7*, 73540–73554. [CrossRef]
2. Adjigble, M.; Tamadazte, B.; de Farias, C.; Stolkin, R.; Marturi, N. 3D Spectral Domain Registration-Based Visual Servoing. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 769–775. [CrossRef]
3. Collewet, C.; Marchand, E. Photometric Visual Servoing. *IEEE Trans. Robot.* **2011**, *27*, 828–834. [CrossRef]
4. Bateux, Q.; Marchand, E. Histograms-Based Visual Servoing. *IEEE Robot. Autom. Lett.* **2017**, *2*, 80–87. [CrossRef]
5. Crombez, N.; Mouaddib, E.M.; Caron, G.; Chaumette, F. Visual Servoing with Photometric Gaussian Mixtures as Dense Features. *IEEE Trans. Robot.* **2019**, *35*, 49–63. [CrossRef]
6. Liang, X.; Wang, H.; Liu, Y.H.; You, B.; Liu, Z.; Jing, Z.; Chen, W. Fully Uncalibrated Image-Based Visual Servoing of 2DOFs Planar Manipulators with a Fixed Camera. *IEEE Trans. Cybern.* **2022**, *52*, 10895–10908. [CrossRef] [PubMed]
7. Liang, X.; Wang, H.; Liu, Y.H.; Chen, W.; Zhao, J. A unified design method for adaptive visual tracking control of robots with eye-in-hand/fixed camera configuration. *Automatica* **2015**, *59*, 97–105. [CrossRef]

8. Brown, J.; Su, D.; Kong, H.; Sukkarieh, S.; Kerrigan, E.C. Improved noise covariance estimation in visual servoing using an autocovariance least-squares approach. *Mechatronics* **2020**, *68*, 102381. [CrossRef]
9. Zhong, X.; Zhong, X.; Hu, H.; Peng, X. Adaptive Neuro-Filtering Based Visual Servo Control of a Robotic Manipulator. *IEEE Access* **2019**, *7*, 76891–76901. [CrossRef]
10. Dong, G.; Zhu, Z.H. Kinematics-based incremental visual servo for robotic capture of non-cooperative target. *Robot. Auton. Syst.* **2019**, *112*, 221–228. [CrossRef]
11. Chaumette, F.; Hutchinson, S. Visual servo control. I. Basic approaches. *IEEE Robot. Autom. Mag.* **2006**, *13*, 82–90. [CrossRef]
12. Siradjuddin, I.; Behera, L.; McGinnity, T.M.; Coleman, S. Image-Based Visual Servoing of a 7-DOF Robot Manipulator Using an Adaptive Distributed Fuzzy PD Controller. *IEEE/ASME Trans. Mechatron.* **2014**, *19*, 512–523. [CrossRef]
13. Anwar, A.; Lin, W.; Deng, X.; Qiu, J.; Gao, H. Quality Inspection of Remote Radio Units Using Depth-Free Image-Based Visual Servo with Acceleration Command. *IEEE Trans. Ind. Electron.* **2019**, *66*, 8214–8223. [CrossRef]
14. Shi, H.; Hwang, K.S.; Li, X.; Chen, J. A learning approach to image-based visual servoing with a bagging method of velocity calculations. *Inf. Sci.* **2019**, *481*, 244–257. [CrossRef]
15. Jin, Z.; Wu, J.; Liu, A.; Zhang, W.A.; Yu, L. Gaussian process-based nonlinear predictive control for visual servoing of constrained mobile robots with unknown dynamics. *Robot. Auton. Syst.* **2021**, *136*, 103712. [CrossRef]
16. Jianhong, W. Dynamic Programming in Data Driven Model Predictive Control. *WSEAS Trans. Syst.* **2021**, *20*, 170–177. [CrossRef]
17. Li, Y.; Li, L.; Zhang, C. AMT Starting Control as a Soft Starter for Belt Conveyors Using a Data-Driven Method. *Symmetry* **2021**, *13*, 1808. [CrossRef]
18. Qiu, Z.; Hu, S.; Liang, X. Disturbance observer based adaptive model predictive control for uncalibrated visual servoing in constrained environments. *ISA Trans.* **2020**, *106*, 40–50. [CrossRef]
19. He, S.; Xu, Y.; Guan, Y.; Li, D.; Xi, Y. Synthetic Robust Model Predictive Control with Input Mapping for Constrained Visual Servoing. *IEEE Trans. Ind. Electron.* **2022**, *70*, 9270–9280. [CrossRef]
20. Qian, J.; Su, J. Online estimation of image Jacobian matrix by Kalman-Bucy filter for uncalibrated stereo vision feedback. In Proceedings of the 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), Washington, DC, USA, 11–15 May 2002; IEEE: Piscataway, NJ, USA, 2002; Volume 1; pp. 562–567.
21. Zhong, X.; Zhong, X.; Peng, X. Robust Kalman filtering cooperated Elman neural network learning for vision-sensing-based robotic manipulation with global stability. *Sensors* **2013**, *13*, 13464–13486. [CrossRef]
22. López-Franco, C.; López-Franco, M.; Alanis, A.Y.; Gómez-Avila, J.; Arana-Daniel, N. Real-Time Inverse Optimal Neural Control for Image Based Visual Servoing with Nonholonomic Mobile Robots. *Math. Probl. Eng.* **2015**, *2015*, 347410. [CrossRef]
23. Gong, Z.; Qiu, C.; Tao, B.; Bai, H.; Yin, Z.; Ding, H. Tracking and grasping of moving target based on accelerated geometric particle filter on colored image. *Sci. China Technol. Sci.* **2021**, *64*, 755–766. [CrossRef]
24. Han, N.; Ren, X.; Zheng, D. Visual servoing control of robotics with a neural network estimator based on spectral adaptive law. *IEEE Trans. Ind. Electron.* **2023**, *70*, 12586–12595. [CrossRef]
25. Wang, F.; Sun, F.; Zhang, J.; Lin, B.; Li, X. Unscented particle filter for online total image Jacobian matrix estimation in robot visual servoing. *IEEE Access* **2019**, *7*, 92020–92029. [CrossRef]
26. He, S.; Xu, Y.; Li, D.; Xi, Y. Eye-in-hand visual servoing control of robot manipulators based on an input mapping method. *IEEE Trans. Control. Syst. Technol.* **2022**, *31*, 402–409. [CrossRef]
27. Zhang, S.; Zhuan, X. Two-Dimensional Car-Following Control Strategy for Electric Vehicle Based on MPC and DQN. *Symmetry* **2022**, *14*, 1718. [CrossRef]
28. Zhang, S.; Zhuan, X. Distributed Model Predictive Control for Two-Dimensional Electric Vehicle Platoon Based on QMIX Algorithm. *Symmetry* **2022**, *14*, 2069. [CrossRef]
29. Urrea, C.; Saa, D. Design, Simulation, Implementation, and Comparison of Advanced Control Strategies Applied to a 6-DoF Planar Robot. *Symmetry* **2023**, *15*, 1070. [CrossRef]
30. Wu, J.; Jin, Z.; Liu, A.; Yu, L. Vision-based neural predictive tracking control for multi-manipulator systems with parametric uncertainty. *ISA Trans.* **2021**, *110*, 247–257. [CrossRef]
31. Klenske, E.D.; Zeilinger, M.N.; Schölkopf, B.; Hennig, P. Gaussian process-based predictive control for periodic error correction. *IEEE Trans. Control Syst. Technol.* **2015**, *24*, 110–121. [CrossRef]
32. Thuruthel, T.G.; Falotico, E.; Renda, F.; Laschi, C. Learning dynamic models for open loop predictive control of soft robotic manipulators. *Bioinspir. Biomim.* **2017**, *12*, 066003. [CrossRef]
33. Abraham, I.; De La Torre, G.; Murphey, T.D. Model-based control using Koopman operators. *arXiv* **2017**, arXiv:1709.01568.
34. Zhu, X.; Ding, C.; Jia, L.; Feng, Y. Koopman operator based model predictive control for trajectory tracking of an omnidirectional mobile manipulator. *Meas. Control* **2022**, *55*, 1067–1077. [CrossRef]
35. Bruder, D.; Fu, X.; Gillespie, R.B.; Remy, C.D.; Vasudevan, R. Data-Driven Control of Soft Robots Using Koopman Operator Theory. *IEEE Trans. Robot.* **2021**, *37*, 948–961. [CrossRef]
36. Tan, N.; Yu, P.; Zheng, W. Uncalibrated and Unmodeled Image-Based Visual Servoing of Robot Manipulators Using Zeroing Neural Networks. *IEEE Trans. Cybern.* **2022**, 1–14. [CrossRef] [PubMed]
37. Zhang, T.; Yan, P. Symmetric Time-Variant IBLF-Based Tracking Control with Prescribed Performance for a Robot. *Symmetry* **2023**, *15*, 1919. [CrossRef]

38. Zheng, Y.X.; Liao, Y. Missile Control Parameters Estimation That Uses Robust Adaptive Kalman Filter Algorithm. In Proceedings of the 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 27–28 August 2016; Volume 1, pp. 226–229. [CrossRef]
39. Korda, M.; Mezić, I. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica* **2018**, *93*, 149–160. [CrossRef]
40. Wang, J.; Xu, B.; Lai, J.; Wang, Y.; Hu, C.; Li, H.; Song, A. An Improved Koopman-MPC Framework for Data-Driven Modeling and Control of Soft Actuators. *IEEE Robot. Autom. Lett.* **2023**, *8*, 616–623. [CrossRef]
41. Yan, X. A Computer Graphic Image Technology with Visual Communication Based on Data Mining. *Wseas Trans. Signal Process.* **2022**, *18*, 89–95. [CrossRef]
42. Zhang, J.; Wang, H. Online Model Predictive Control of Robot Manipulator With Structured Deep Koopman Model. *IEEE Robot. Autom. Lett.* **2023**, *8*, 3102–3109. [CrossRef]
43. Mamakoukas, G.; Abraham, I.; Murphey, T.D. Learning Stable Models for Prediction and Control. *IEEE Trans. Robot.* **2023**, *39*, 2255–2275. [CrossRef]
44. Zhang, X.; Pan, W.; Scattolini, R.; Yu, S.; Xu, X. Robust tube-based model predictive control with Koopman operators. *Automatica* **2022**, *137*, 110114. [CrossRef]
45. Qiu, Z.; Wu, Z. Adaptive neural network control for image-based visual servoing of robot manipulators. *IET Control Theory Appl.* **2022**, *16*, 443–453. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.