Vision-Based Robotic Control

Wei Qi Yan

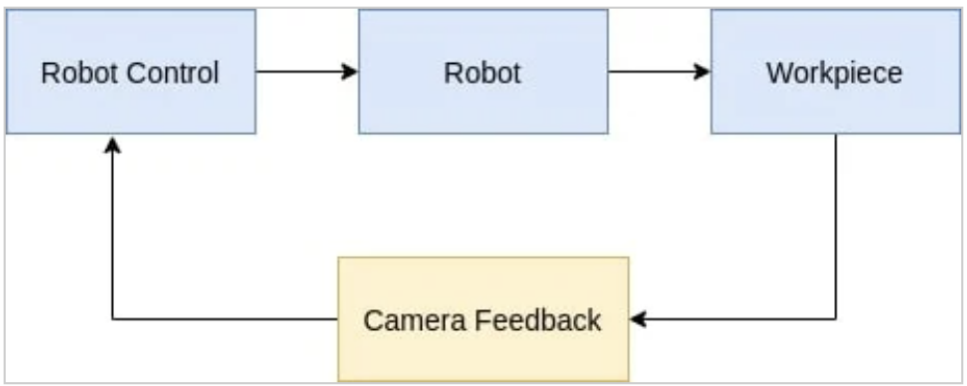
Auckland University of Technology

Table of Contents

Vision-Based Robotic Control

Vision-Based Robot Control

- Robot manipulators (i.e., robot arms) are extensively deployed in manufacturing, packaging, and processing factories.
- Visual servoing is the method of controlling a robot's motion using real-time feedback from vision sensors to execute tasks.
- Visual servoing is a model-free approach to actuate the robot based on the reference high-level task to be executed.



https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

Vision-Based Robot Control

In visual servoing, the robot is instructed to move in order to align its current task progress metrics with the desired task metrics and gradually reduce the error between the two.

Mathematically, visual servoing is to minimize the error:

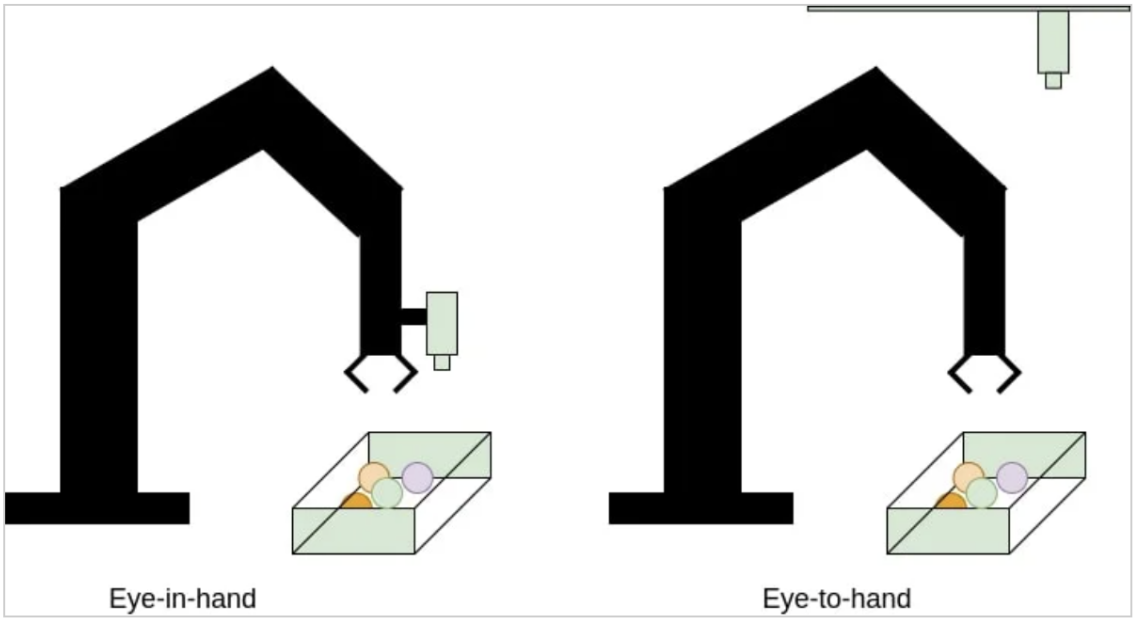
$$e(t) = f(m(t), a) - f^*(m(t), a)$$

where $m(t)$ is the collection of regions of interest in the image and a is the collection of camera intrinsic and extrinsic parameters. The function $f^*(\cdot)$ represents the desired set of visual features while $f(\cdot)$ reflects the actual visual features.

https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

Vision-Based Robot Control

Depending on the positioning of the camera, visual servoing has two paradigms: Eye-in-hand and eye-to-hand.



The camera is mounted on the end-effector of the robot.

Vision-Based Robotic Control

Vision-Based Robot Control

- Visual servoing is to control the pose of robot's end-effector using visual features extracted from the image which includes two approaches: Position/Pose-Based Visual Servo (PBVS) and Image-Based Visual Servo (IBVS).
- PBVS takes use of observed visual features, a calibrated camera and a known geometric model of the target to determine the pose of target with respect to the camera.
- IBVS omits the pose estimation step, and adopts the image features directly. The desired camera pose with respect to the target is defined implicitly by using the image features at the goal pose.

https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

Vision-Based Robotic Control

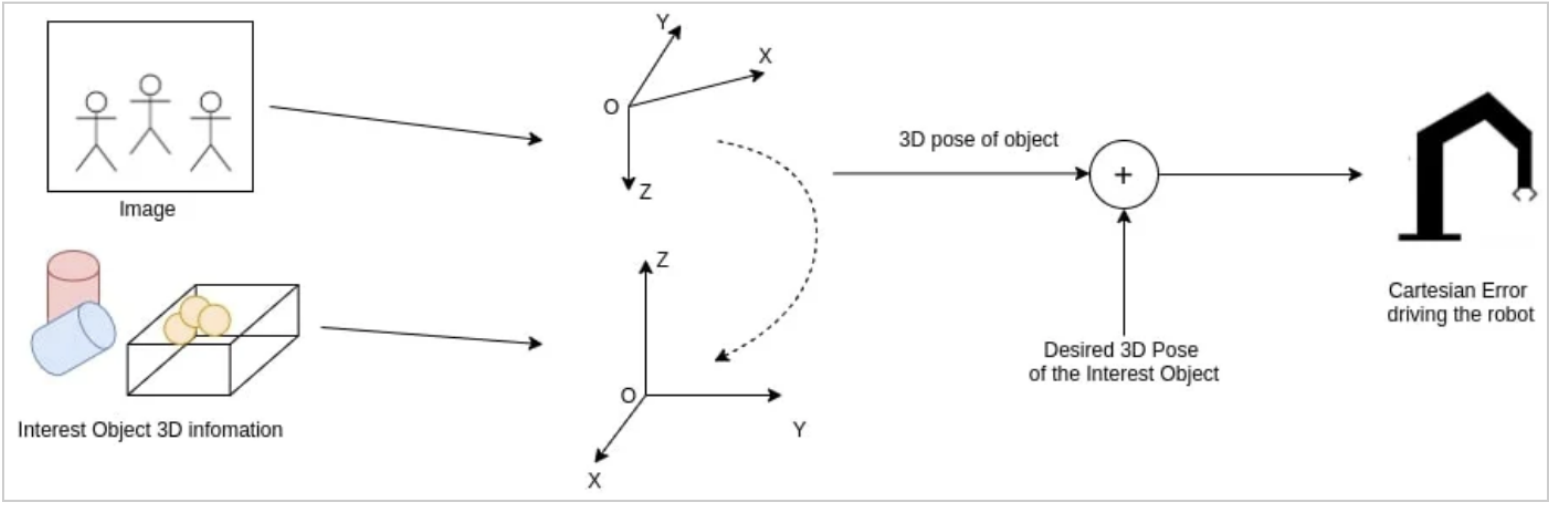
PBVS: Position/Pose-Based Visual Servoing

- PBVS usually uses depth cameras to obtain the 3D pose/position and orientation of the regions/objects of interest.
- The error term is the Cartesian pose difference between the two.
- The servoing scheme is to minimize it by moving the robot around, ideally towards the final desired pose.
- Based on the location of the object in the image, the scheme generates the ideal grasp pose for the end-effector and converge the robot to it.

https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

PBVS: Position/Pose-Based Visual Servoing

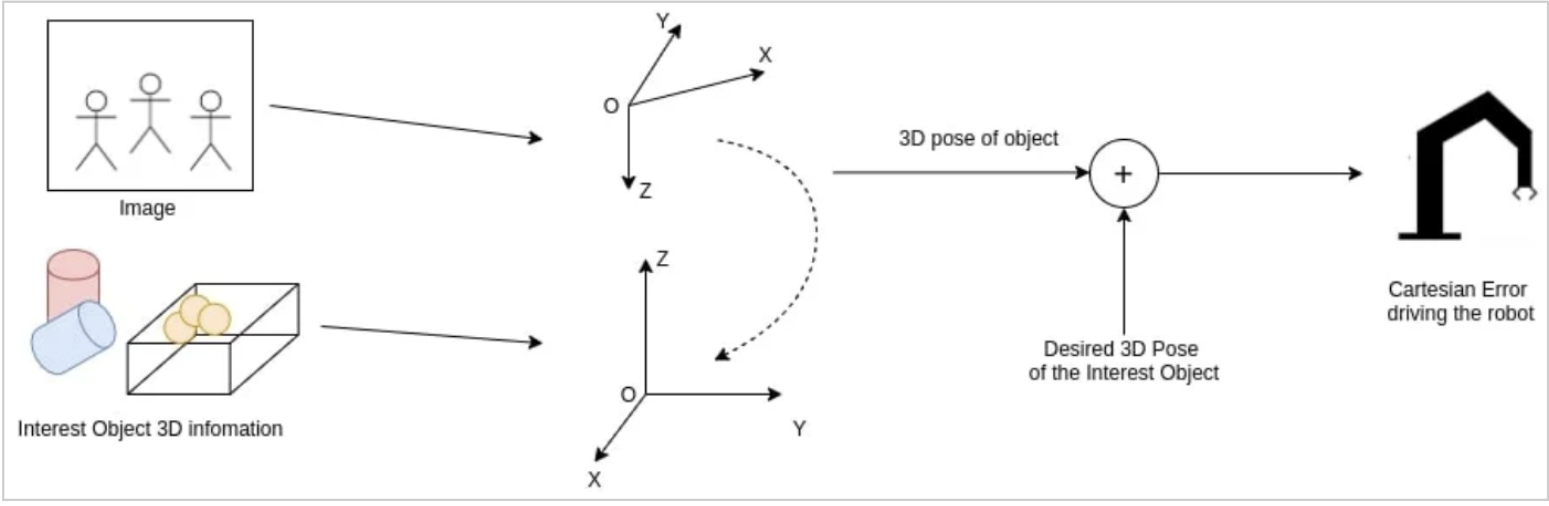
PBVS works with real-world poses which needs at least a 6-DOF robot arm to successfully implement the solution without getting stuck in local minima.



https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

PBVS: Position/Pose-Based Visual Servoing

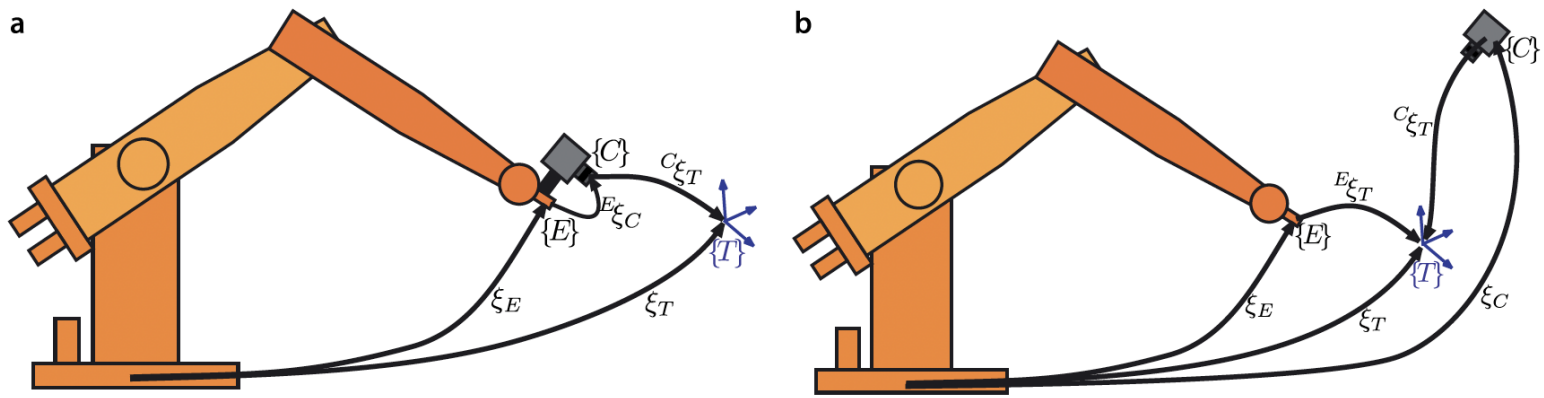
PBVS makes use of robot inverse kinematics to convert Cartesian control instructions into joint angle values for the robot.



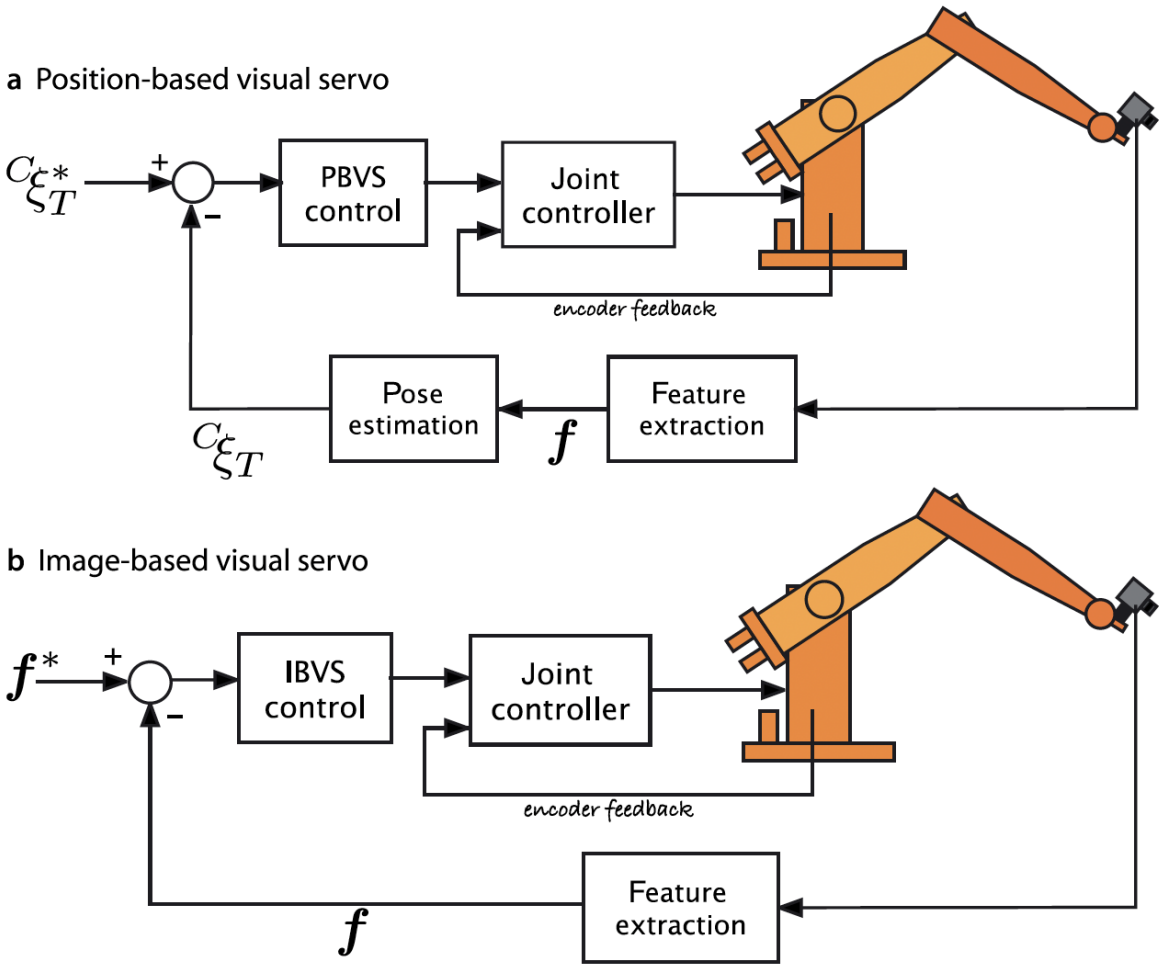
https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

PBVS: Position/Pose-Based Visual Servoing

Since obtaining the 3D pose involves the conversion of information from the camera frame to the robot frame, camera calibration plays a significant role in the PBVS process.

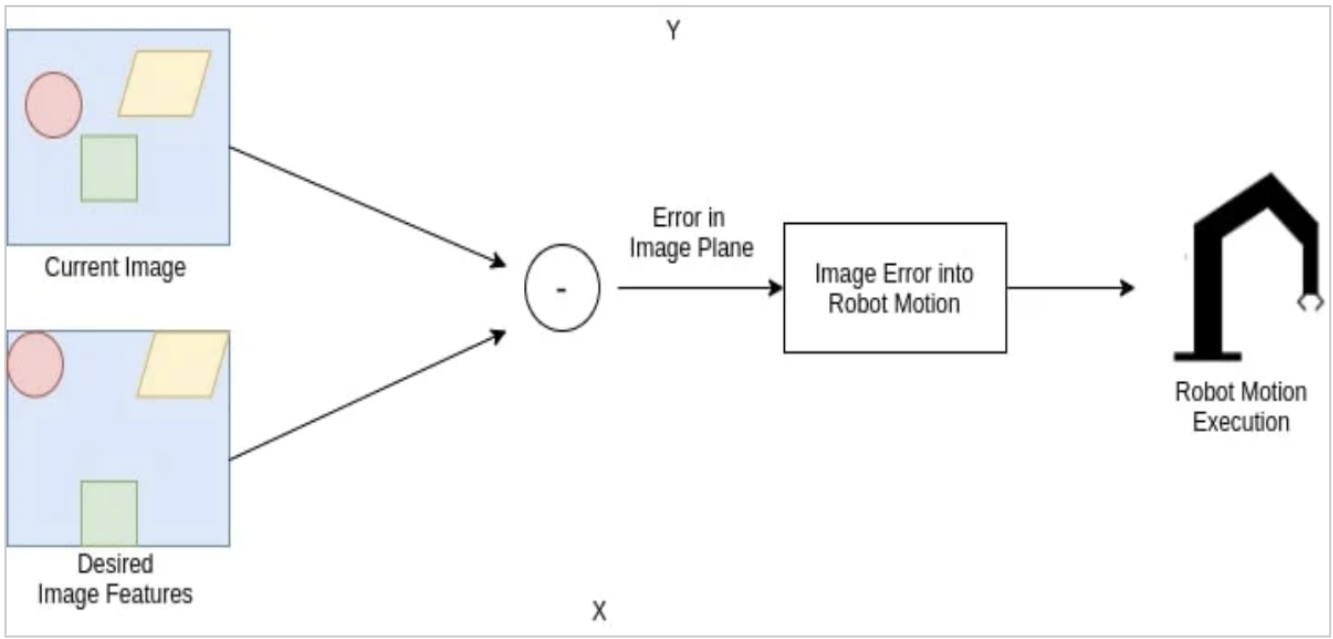


PBVS & IBVS


a Position-based visual servo
$C\xi_T^*$
+
–
PBVS control
Joint controller
encoder feedback
Pose estimation
Feature extraction
$C\xi_T$
f
b Image-based visual servo
f^*
+
–
IBVS control
Joint controller
encoder feedback
Feature extraction
f

IBVS: Image-Based Visual Servoing

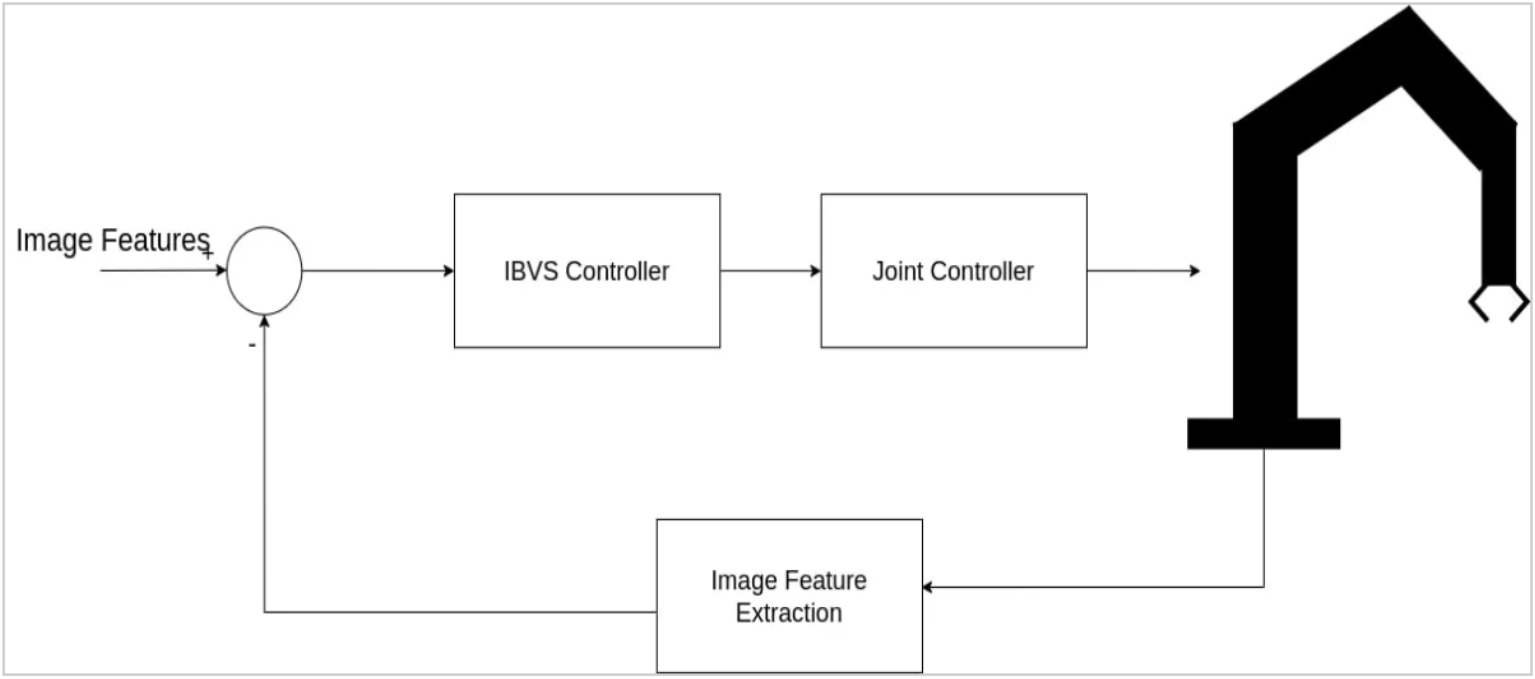
- IBVS extracts visual features and then formulates the error in the image plane itself.
- The visual servoing scheme converges visual feature to the desired coordinates and moves the robot accordingly in image space.



https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

IBVS: Image-Based Visual Servoing

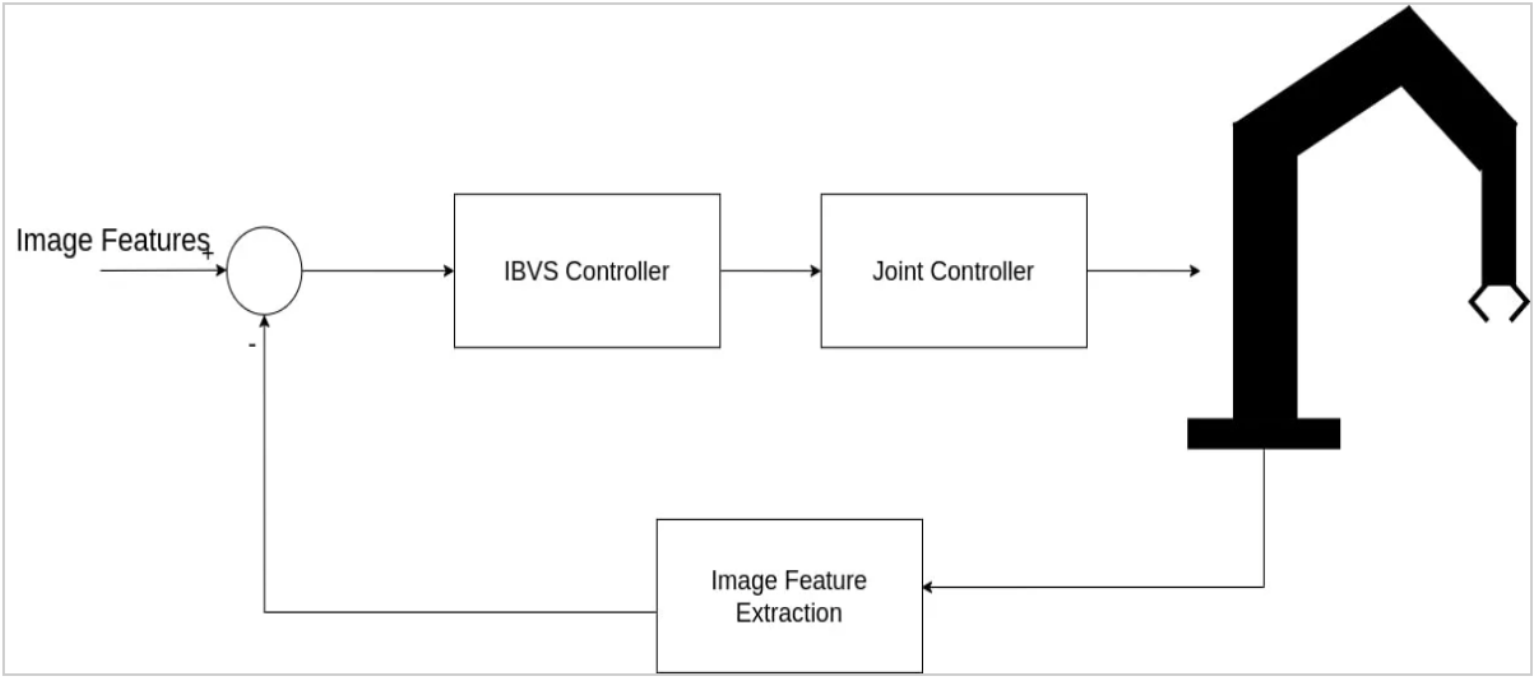
The visual feature extraction in IBVS is prone to camera performance, synchronization issues, and computational requirements.



https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

IBVS: Image-Based Visual Servoing

The first step of IBVS is the projection of the 3D object on a 2D image plane: $x = X/Z$ and $y = Y/Z$.



https://control.com/technical-articles/an-overview-of-visual-servoing-for-robot-manipulators/

IBVS: Image-Based Visual Servoing

- IBVS differs fundamentally from PBVS by not estimating the relative pose of the target.
- The relative pose is implicit in the values of the image features.
- IBVS is a remarkably robust approach to vision-based robot control.
- IBVS can also be formulated to work with other image features such as line, circle, rectangle, as found by Hough transform.

Questions?

Questions?

IBVS differs from PBVS:

1. by estimating the relative pose of the target.
2. by not estimating the relative pose of the target.
3. by averaging the relative pose of the target.
4. None of the given options.

The right answer is:___.

Questions?

Vision-Based Robotic Control

Visual Servoing

- Autonomous robot operation is proposed to resolve the lack of labor issue in industries and visual servoing is one of the most important technologies for this issue.
- Industries often need robots to be running at lightning-fast speeds, visual servoing is far from achieving speedy performances.
- The computational bottlenecks in image processing and inverse kinematics are on using GPUs and parallel programming.

Camera Retreat

The camera retreat problem refers to the camera that needs to be moved back (or "retreated") to capture an entire scene or object within the view frustum (i.e., the visible area of the scene).

The camera retreat problem can be resolved by:

- Adjusting the camera position;
- Changing the Field of View (FOV);
- Scaling the object;
- Adjusting clipping planes.

XY/Z-Partitioned IBVS

- We encounter the problem of camera retreat (moving back) in an IBVS system because the object is too large or too close.
- For a rotating camera, the points will naturally move along circular arcs.
- The scale change is achieved by z-axis translation.
- The XY/Z hybrid schemes consider the x-axis and y-axis as one group, and the z-axis as another group.

XY/Z-Partitioned IBVS

- XY/Z-partitioned methods eliminate camera retreat by using IBVS to control the Degrees of Freedom (DoF) while using a different controller for the remaining degrees of freedom.
- It is advantageous to select the longest line segment that can be constructed from the feature points, which may be changed during the motion as the feature point configuration changes.
- The desired rotational rate is obtained by using a simple proportional control law.
- We project features from one or more cameras of any type onto spherical image plane, and compute the control law in terms of spherical coordinates.

IBVS Using Polar Coordinates

- In polar coordinates, the image point is written as $p = (r, \phi)$ where r is the distance of the point from the principal point, $r = \sqrt{u^2 + v^2}$, where we recall that u and v are the image coordinates with respect to the principal point rather than the image origin.
- The angle from the u-axis to a line joining the principal point to the image point is $\phi = tan^{-1}(\frac{v}{u})$.
- The two coordinate representations are related by $u = r\cos(\phi)$ and $v = r\sin(\phi)$.
- The performance of polar IBVS is the complement of Cartesian IBVS.

IBVS for a Spherical Camera

- The world point $\mathbf{P} = (X, Y, Z)$ in the camera frame is projected onto the surface of the sphere at the point $p = (x, y, z)$, $x = X/R, y = Y/R, z = Z/R$, R is the distance from the camera origin to the world point.
- A minimal spherical coordinate system comprises of the angle of colatitude $\theta = \sin^{-1}(y/r)$, $\theta \in [0, \pi]$, $r = \sqrt{x^2 + y^2}$.
- The point vector $\mathbf{p} = (\theta, \phi)$, $\phi = sin^{-1}(\frac{z}{r})$.
- We obtain $X = R \cdot \cos(\theta)\cos(\phi)$, $Y = R \cdot \cos(\theta)\sin(\phi)$, $Z = R \cdot \sin(\theta)$, $R = \sqrt{X^2 + Y^2 + Z^2}$.

IBVS for a Spherical Camera

Spherical imaging has advantages for visual servoing:

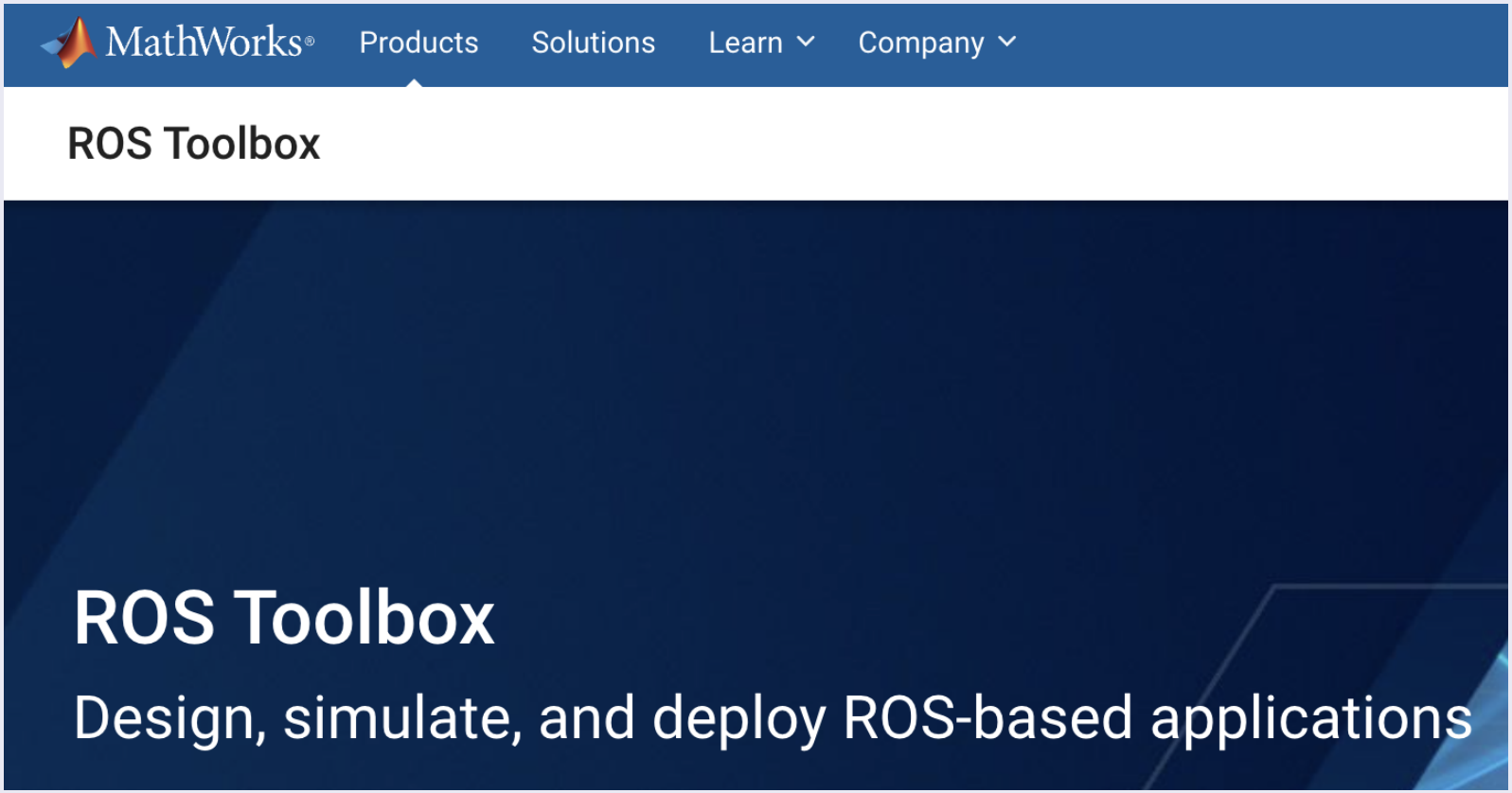
- A spherical camera eliminates the need to explicitly keep features in the field of view which is a problem with both position-based visual servoing and hybrid schemes.
- For IBVS with a long focal length, this can lead to slow convergence and/or sensitivity to noise in feature coordinates.
- For a spherical camera, with the largest possible field of view, this ambiguity is reduced.
- We project images from one or more cameras onto spherical image plane, and compute the control law in terms of spherical coordinates.

Applications

- In 6-axis arm-type robot, a perspective camera with default parameters is mounted on the robot's end-effector and its axes are aligned with the coordinate frame of the robot's end-effector. This system drives the robot to the desired pose.
- In mobile robot, we consider a camera mounted on a mobile robot moving in a planar environment, the visual servo controller will drive the robot until its view of the landmarks matches the desired view.

Questions?

MATLAB ROS Toolbox



https://au.mathworks.com/products/ros.html

MATLAB ROS Toolbox

- Robot Operating Systems (ROS) are frameworks or collections of software libraries and tools that help developers create robotic applications.
- ROS is a middleware that provides services such as hardware abstraction, device control, message passing between processes, and package management.

https://au.mathworks.com/products/ros.html

MATLAB ROS Toolbox

- **Modularity**: ROS breaks down complex robotic systems into manageable components called nodes. Each node performs a specific task and communicates with other nodes using topics, services, or actions.
- **Hardware Abstraction**: ROS provides a layer of abstraction between hardware and software, developers will not worry about the underlying hardware specifics.
- **Communication**: ROS provides a flexible and efficient communication infrastructure which can happen within the same machine or across multiple machines on a network.
- **Tools**: ROS comes with a suite of powerful tools for debugging, visualization, and simulation.
- **Package Management**: ROS organizes code into packages, which can be easily shared and reused, easy integration of third-party software.

https://au.mathworks.com/products/ros.html

MATLAB ROS Toolbox

- ROS 2 is the second generation of the Robot Operating System (ROS).
- ROS 2 is designed with real-time performance.
- ROS 2 uses the Data Distribution Service (DDS) as the communication framework that enables reliable, real-time, and scalable communication between distributed systems.
- ROS 2 introduces enhanced security features, including secure communication and data encryption.
- ROS 2 is better suited for coordinating multiple robots working together.
- ROS 2 offers better support for multiple operating systems, including Microsoft Windows and Apple macOS.
- ROS 2 provides improved tools for testing, debugging, and monitoring robotic systems.

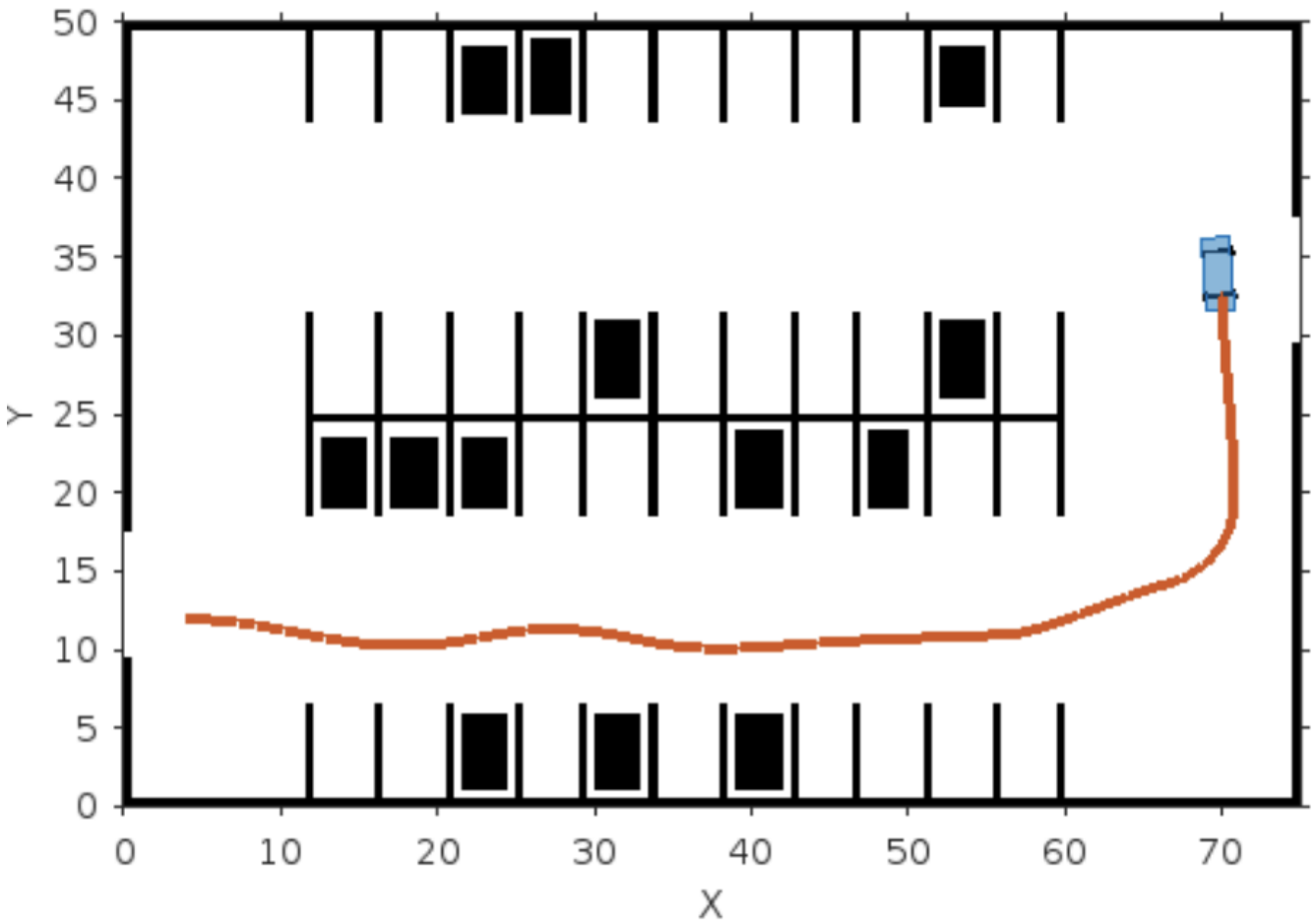
https://au.mathworks.com/products/ros.html

MATLAB ROS Toolbox

- ROS Toolbox provides an interface connecting MATLAB and Simulink with the Robot Operating System (ROS and ROS 2).
- With the toolbox, we design a network of ROS nodes and combine MATLAB or Simulink generated ROS nodes with the existing ROS network.
- The toolbox includes MATLAB functions to visualize and analyze ROS data by recording, importing, and playing back ROS files.
- The toolbox lets us verify ROS nodes via desktop simulation and by connecting to external robot simulators.

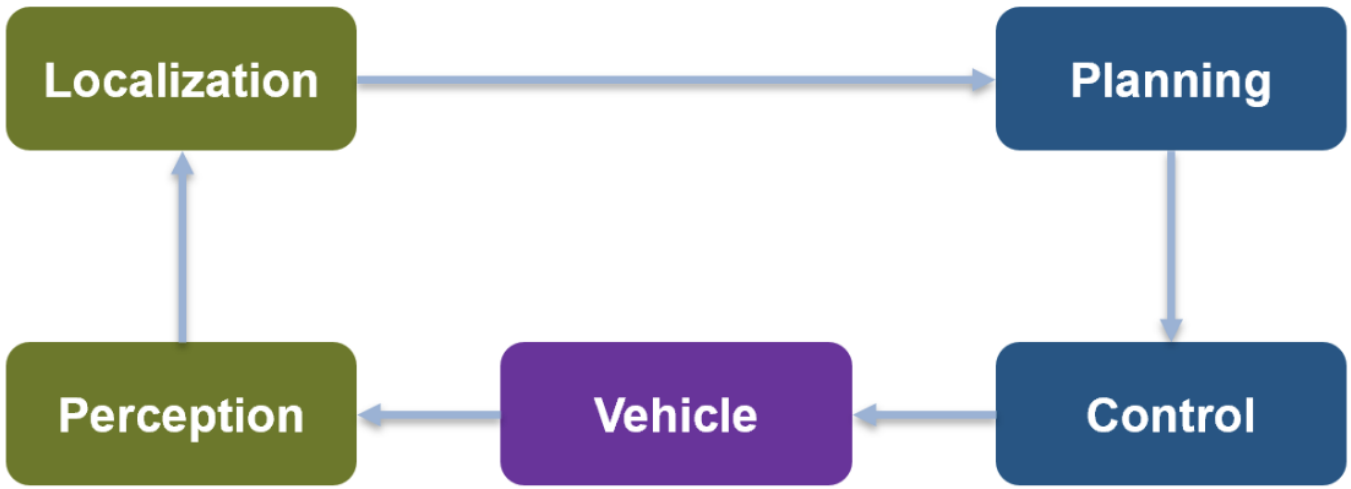
https://au.mathworks.com/products/ros.html

MATLAB ROS Toolbox



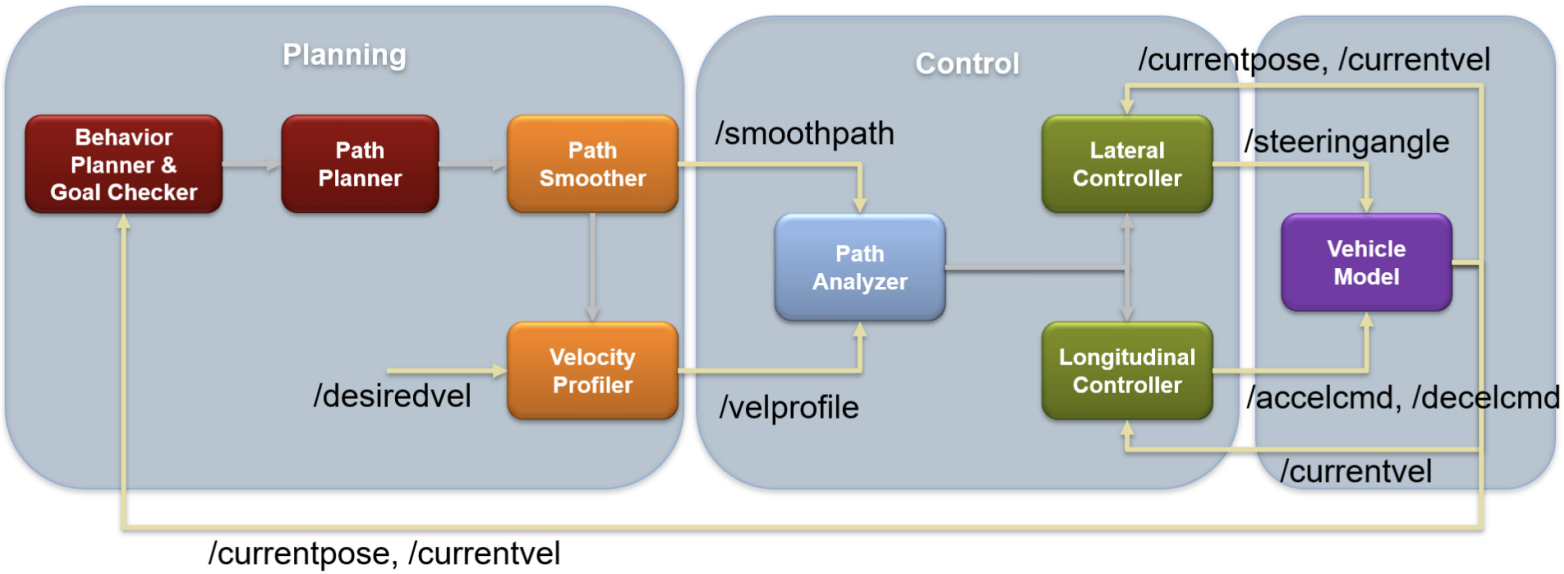
https://au.mathworks.com/help/ros/ug/ros-automated-valet.html

MATLAB ROS Toolbox



https://au.mathworks.com/help/ros/ug/ros-automated-valet.html

MATLAB ROS Toolbox



https://au.mathworks.com/help/ros/ug/ros-automated-valet.html

Questions?

Questions?

In camera retreat, the view frustum refers to:

1. The visible area of the scene.
2. The field of view (FoV).
3. A truncated cone
4. A truncated pyramid

The right answer is:___.

Questions?

Learning Objectives

- Derive solutions for particular robotic vision and visual control tasks characterised by specifics of image data and deep learning algorithms.
- Critically evaluate the performance of robotic vision with deep learning algorithms, bench mark data, performance measures, and ways to define ground truth.