“To infinity ... and beyond!”

Buzz Lightyear
Vision-based Mobile-Robot Navigation

CSE 4392-5369 “Vision-based Robot Sensing, Localization and Control”

Dr. Gian Luca Mariottini, Ph.D.

Department of Computer Science and Engineering
University of Texas at Arlington

WEB: http://ranger.uta.edu/~gianluca/
Suppose given a robot that can measure its **distance and orientation with respect to a wall**.

- The **distance to the wall** is given by $\delta$
- The **orientation wrt the wall** is given by $\varphi_R - \varphi^*$

Our **goal** is to devise a control strategy $[w,w]$ that zeroes the distance and orientation errors:

$$e_1(t) \triangleq \delta(t) - \delta^*$$
$$e_2(t) \triangleq \varphi_R(t) - \varphi^*$$

Note that:

$$\varphi^* = \theta - \pi/2$$

As seen in previous classes:

$$\delta(t) = x(t) \cos \theta + y(t) \sin \theta - \rho$$
An example: Wall-Following (2)

The robot changes its pose according to the following equations:

\[
\begin{align*}
\dot{x}(t) &= v_R(t) \cos \varphi_R(t) \\
\dot{y}(t) &= v_R(t) \sin \varphi_R(t) \\
\dot{\varphi}_R(t) &= \omega_R(t)
\end{align*}
\]

By differentiating wrt time the expressions \((e_1, e_2)\)

\[
\begin{align*}
\dot{e}_1(t) &= \dot{\delta}(t) = v_R(t) \cos(\varphi_R(t) - \theta) \\
\dot{e}_2(t) &= \dot{\varphi}(t) = \omega_R(t)
\end{align*}
\]

So we have:

\[
\dot{e}(t) = \begin{bmatrix} \dot{e}_1(t) \\ \dot{e}_2(t) \end{bmatrix} = \begin{bmatrix} \cos(\varphi_R(t) - \theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_R \\ \omega_R \end{bmatrix}
\]
An example: Wall-Following (3)

And by imposing exponential behavior we have:

\[ \dot{e}(t) = -\lambda e(t) \]

where:

\[ \dot{e}(t) = \begin{bmatrix} \dot{e}_1(t) \\ \dot{e}_2(t) \end{bmatrix} = \begin{bmatrix} \cos(\varphi_R(t) - \theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_R \\ \omega_R \end{bmatrix} \]

So we have:

\[ -\lambda e(t) = \begin{bmatrix} \cos(\varphi_R(t) - \theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_R \\ \omega_R \end{bmatrix} \]

\[ \triangleq A \]

\[ \begin{bmatrix} v_R \\ \omega_R \end{bmatrix} = -\lambda A^{-1}(t) e(t) \]

(See Ex_WallFollowing.m)
Problem:
Plan a path that leads a mobile robot (unicycle) from an initial configuration \( \mathbf{x}_i \triangleq [x_i, y_i, \theta_i]^T \) to a final configuration \( \mathbf{x}_f \triangleq [x_f, y_f, \theta_f]^T \).

Approach:
Find a curve passing from \( [x_i, y_i]^T \) and \( [x_f, y_f]^T \) with a curvature which renders it compatible with the unicycle model.

\[
\begin{align*}
\dot{x}(t) &= v_R(t) \cos \theta(t) \\
\dot{y}(t) &= v_R(t) \sin \theta(t) \\
\dot{\theta}(t) &= \omega_R(t)
\end{align*}
\]

If a 2-D map of the area is known (up to scale) then the current view of the field can be used to estimate the homography, thus we can plan the trajectory back on the map.

Note that the path planned using trajectory-planning will be given as input to the motion control algorithm.
The curve we are looking for is parameterized with the curve parameter $s$

\[
\begin{align*}
\left[\begin{array}{c}
x(s) \\
y(s)
\end{array}\right] &= \begin{cases} 
\left[\begin{array}{c}
x_i \\
y_i
\end{array}\right] & \text{if } s = 0 \\
\left[\begin{array}{c}
x_f \\
y_f
\end{array}\right] & \text{if } s = 1
\end{cases}
\end{align*}
\]

\[
x(s) = s^3 x_f - (s - 1)^3 x_i + \alpha_x s^2 (s - 1) + \beta_x s (s - 1)^2
\]

\[
y(s) = s^3 y_f - (s - 1)^3 y_i + \alpha_y s^2 (s - 1) + \beta_y s (s - 1)^2
\]

The above two expressions satisfy the boundary conditions in (i).

Regarding the orientation $\theta(s)$ this can be obtained with:

\[
\begin{bmatrix}
\dot{x} \\
\dot{y}
\end{bmatrix} = \begin{bmatrix}
\frac{\partial x}{\partial t} \\
\frac{\partial y}{\partial t}
\end{bmatrix} = \begin{bmatrix}
\frac{\partial x}{\partial s} \dot{s} \\
\frac{\partial y}{\partial s} \dot{s}
\end{bmatrix} = \begin{bmatrix}
x'(s) \\
y'(s)
\end{bmatrix}
\]

so that:

\[
\theta(s) = \text{Atan2}(y'(s), x'(s)) + k\pi, \quad k \in \{0, 1\}
\]
Trajectory planning via Cartesian Polynomials (3)

We need to find the values of $\alpha_x, \beta_x, \alpha_y, \beta_y$

\[
x'(s) = 3s^2 x_f - 3(s - 1)^2 x_i + \alpha_x [2s(s - 1) + s^2] + \beta_x [(s - 1)^2 + 2s(s - 1)]
\]

\[
y'(s) = 3s^2 y_f - 3(s - 1)^2 y_i + \alpha_y [2s(s - 1) + s^2] + \beta_y [(s - 1)^2 + 2s(s - 1)]
\]

And by imposing now the boundary conditions and the unicycle kinematic:

\[
x'(0) = -3x_i + \beta_x = k_i \cos \theta_i \quad x'(1) = 3x_f + \alpha_x = k_f \cos \theta_f
\]

\[
y'(0) = -3y_i + \beta_y = k_i \sin \theta_i \quad y'(1) = 3y_f + \alpha_y = k_f \sin \theta_f
\]

\[
\beta_x = k_i \cos \theta_i + 3x_i \quad \alpha_x = k_f \cos \theta_f + 3x_f
\]

\[
\beta_y = k_i \sin \theta_i + 3y_i \quad \alpha_y = k_f \sin \theta_f + 3y_f
\]

For two given values of $k_i$ and $k_f$, then we can compute the values of $\alpha_x, \beta_x, \alpha_y, \beta_y$ and, thus, the values of the final trajectory (see f_trajectoryplan.m)
Navigation based on Input/Output Linearization (1)

We consider the following measurements (outputs) \( \mathbf{y} = [y_1 \ y_2]^T \)

\[
y_1(t) = x(t) + b \cos \theta(t)
\]

\[
y_1(t) = x(t) + b \sin \theta(t)
\]

We are looking for a control that minimizes the error:

\[
\mathbf{e}(t) = \begin{bmatrix}
y_1(t) - y_{1}^{des} \\
y_2(t) - y_{2}^{des}
\end{bmatrix}
\]

where it results:

\[
\dot{\mathbf{y}} = \begin{bmatrix}
\dot{y}_1 \\
\dot{y}_2
\end{bmatrix} = \begin{bmatrix}
x - b\dot{\theta} \sin \theta \\
\dot{y} + b\dot{\theta} \cos \theta
\end{bmatrix} = \begin{bmatrix}
v_R(t) \cos \theta(t) - b \omega_R(t) \sin \theta(t) \\
v_R(t) \sin \theta(t) + b \omega_R(t) \cos \theta(t)
\end{bmatrix}
\]

that is:

\[
\dot{\mathbf{y}} = T_{\theta(t)} \mathbf{u}
\]

and we define:

\[
\mathbf{e} \triangleq \mathbf{y} - \mathbf{y}^{des}
\]

\[
\dot{\mathbf{e}} \triangleq \dot{\mathbf{y}} - \dot{\mathbf{y}}^{des}
\]
Navigation based on Input/Output Linearization (2)

- So that, by imposing an exponential decrease to zero of the error
  \[
  \dot{e} = -Ke
  \]
  and being \( e \equiv y - y^{des} \) and \( \dot{e} \equiv \dot{y} - \dot{y}^{des} \)
  \[
  \dot{y} - \dot{y}^{des} = -K(y - y^{des})
  \]

- So that we obtain:
  \[
  T_{\theta(t)} u = \dot{y}^{des} - K(y - y^{des})
  \]

- And finally:
  \[
  u = T_{\theta(t)}^{-1} \left( \dot{y}^{des} - K(y - y^{des}) \right)
  \]
  Note that this time the inverse of matrix \( T_{\theta(t)} \) exists always.
Navigation based on Input/Output Linearization (3)

Gian Luca Mariottini, Univ. Texas at Arlington
PBVS describes the task in terms of Cartesian pose

PBVS is very sensitive to **calibration error** (with which $c_x$ is computed)

PBVS needs an **accurate model of the observed scene** (e.g., CAD model)
Image-based Visual Servoing [Espiau et al., TRA'92]

- IBVS is defined in terms of image features coordinates
- Robust to both camera and robot calibration error

\[
L(f, Z)
\]

Convergence is local around the desired position

- Difficult to analyze the stability w.r.t. calibration errors
- The Interaction Matrix depends on the depth \( Z \)
**Hybrid 2-1/2-D Visual Servoing** [Malis et al., TRA'99]

- Does not need any 3-D model of the scene
- Convergence in the whole task space

Camera calibration is needed with nonholonomic mob.rob. [Dixon et al.TAC’05]

Rotation and (scaled) translations are estimated from scene points

Only planar feature points are used [Malis et al., TRA’02]
In this talk: Multiple-View IBVS

- Does not need any 3-D model of the scene
- Convergence in the whole task space

Multiple view image features are used ....no need for \((R, \alpha t)\)

Feature points are not necessarily planar

Partially calibrated camera (in the pinhole camera case)
**IBVS for nonholonomic robots** (pinhole camera)

The **epipoles** are the main features of our IBVS strategy for nonholonomic mobile robots...

**why are they good?**

- **We do not need any 3-D model** of the scene
- **Computed from feature points** (not necessarily co-planar)

- **Multiple view image features** are used ...no need for \((R, \alpha t)\)

- **Global asymptotic convergence** also with **partially uncalibrated camera**

Gian Luca Mariottini, Univ. Texas at Arlington
The nonholonomic visual servoing problem

- Unicycle-like robot moving on a plane;
- \( q = [x \ y \ \theta]^T \) Current robot configuration vector;
- \( q' = [0 \ 0 \ \pi/2]^T \) Desired configuration;
- The desired camera frame coincides with \( \{O, x, y\} \)

Consider this particular **two view** situation, as follows:

The epipole \( e = [e_u, e_v]^T \) is at the image center \( e = [0, 0]^T \) (and \( e', \) too).
Sketch of the control method

1st Step: Zeroing the epipoles \((e_u, e'_u)\)

2nd Step: Matching the features
IBVS 1st Step: Epipole kinematics

\[ e_u = f \frac{x \sin \theta - y \cos \theta}{x \cos \theta + y \sin \theta} \]

\[ e'_u = f \frac{x}{y} \]

and due to the nonholonomic constraints it results:

\[ \dot{e}_u = -u_1 \frac{\text{sign}(e_u e'_u)}{d} \frac{e_u \sqrt{e_u^2 + f^2}}{f} + u_2 \frac{e_u^2 + f^2}{f} \]

\[ \dot{e'}_u = -u_1 \frac{\text{sign}(e_u e'_u)}{d} \frac{e_u (f^2 + e'_u^2)}{f \sqrt{e_u^2 + f^2}}. \]
It is possible to perform an approximate I/O linearization, by setting:

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{\theta}
\end{bmatrix}
= D
\begin{bmatrix}
u_1 \\
u_2
\end{bmatrix}
\]

with

\[
\hat{D}^{-1} =
\begin{bmatrix}
0 & -\frac{\text{sign}(e_u e'_u)}{e_u (e'^2_u + f^2)} \\
\frac{f}{e'^2_u + f^2} & \frac{\hat{d}_f \sqrt{e'^2_u + f^2}}{e_u (e'^2_u + f^2)} \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
\dot{\nu}_1 \\
\dot{\nu}_2
\end{bmatrix}
= \hat{D}^{-1}
\begin{bmatrix}
\nu_1 \\
\nu_2
\end{bmatrix}
\]

NLTV (not decoupled)

LTV (decoupled)
Proposition 1. Let

\[
\begin{bmatrix}
\nu_1 \\
\nu_2
\end{bmatrix} = \begin{bmatrix}
-k_1 e_u \\
-k_2 e_u^{\frac{\beta}{\gamma}}
\end{bmatrix}
\]

where $k_1 > 0$, $k_2 > 0$ and $\beta, \gamma$ are positive odd integers, with $\beta < \gamma$. Also, update the distance estimate $\hat{d}$ according to the following

\[
\dot{\hat{d}} = k_2 f^2 \hat{d} \frac{e_u^{\frac{\beta}{\gamma}}}{e_u (e_u^2 + f^2)}
\]

initialized at $\hat{d}_0 \geq d_0$ where $d_0$ is the (unknown) initial value of $d$. Then, for sufficiently small $k_2$, the approximately linearizing control drives the epipole coordinates $e_u, e'_u$ to zero, for any initial condition, with exponential convergence rate.
Sketch of the Proof. The proposed control law leads us to write the epipole kinematics as:

\[
\dot{e}_u = -k_1 e_u - \left( \frac{\hat{d}}{d} - 1 \right) \frac{e_u^2 + f^2}{e_u'^2 + f^2} k_2 e_u'^{\beta/\gamma} \quad (a)
\]

\[
\dot{e}_u' = -\frac{\hat{d}}{d} k_2 e_u'^{\beta/\gamma} \quad \text{ zeroes in finite time } (b)
\]

and \( d = (x^2 + y^2)^{1/2} \) evolves as:

\[
\dot{d} = k_2 f^2 \frac{e_u'^{\beta/\gamma}}{e_u(e_u'^2 + f^2)}
\]

Then \( \dot{d} \) and \( \hat{d} \) obey to the same differential equations and this yields: \( \frac{\hat{d}}{d} \geq 1 \)

The coefficient in (b) is <0 and bounded below (in modulus): then, 0 is a terminal attractor* for \( e_u' \) (which will converge to zero at \( \bar{t} \)).

From \( \bar{t} \) on, (a) will converge to 0 with exponential rate \( k_1 \). □

* [M. Zak, NN’89]
The **input velocities** can be then written as:

\[
\begin{align*}
    u_1 &= \text{sign}(e_u e'_u) \frac{\hat{d} f \sqrt{e_u^2 + f^2}}{e_u (e_u^2 + f^2)} k_2 e' u^\gamma \\
    u_2 &= -\frac{f}{e_u^2 + f^2} k_1 e_u + \frac{f}{e_u^2 + f^2} k_2 e' u^\gamma 
\end{align*}
\]

- The **zero dynamics** associated to our controller are \( \dot{d} = 0 \) (the robot converges to the \( y \)-axis at a finite distance \( d \)).

- It may be shown that the maximum increment \( \Delta \) on the distance is given by

\[
\Delta = \exp \left( \frac{1}{1 - \beta / \gamma} \frac{|e'_{u,0}|}{|e_{u,\min}|} \right)
\]

- \( \hat{d}_0 \geq d_0 \) can be chosen from a **rough** knowledge of the environment.
**IBVS 2\textsuperscript{nd} Step: Matching the features**

**Basic idea:** translate the robot until each image feature matches the corresponding in the desired view

\[ S = \| p \|^2 - \| p' \|^2 \rightarrow 0 \quad \text{iff} \quad q = q' \]

**Proposition 2.** Let the robot velocities during the second step be defined as

\[ u_1 = -k_t S \]
\[ u_2 = 0 \]

with \( k_t > 0 \). Then, the robot configuration converges exponentially from the intermediate configuration to the origin.
Proposition 3. Let the auxiliary controls $\nu_1, \nu_2$ be chosen as in Proposition 1, and update the distance estimate $\hat{d}$ using $\hat{f}$ for $f$, and according to

$$\dot{\hat{d}} = k_2 \hat{f}^2 \hat{d} \frac{e_u' \beta/\gamma}{e_u (e_u'^2 + \hat{f}^2)}$$

initialized at $\hat{d}_0 \geq d_0$. If $\hat{f} \leq f$, the approximately control

$$\begin{bmatrix}
u_1 \\
u_2 
\end{bmatrix} = \tilde{D}^{-1} \begin{bmatrix} \nu_1 \\
u_2 
\end{bmatrix}$$

drives the epipole coordinates to zero, with exponential convergence rate:

$$\tilde{D}^{-1} = \begin{bmatrix} 0 & -\text{sign}(e_u e'_u) \frac{\hat{f}\sqrt{e_u^2 + \hat{f}^2}}{e_u(e_u'^2 + \hat{f}^2)} \\
\hat{f}/e_u^2 + \hat{f}^2 & -\hat{f}/e_u'^2 + \hat{f}^2 
\end{bmatrix}$$
Realized using the Epipolar Geometry Toolbox (EGT)*

First Step: calibrated camera

\[ k_1 = 0.5, \quad k_2 = 0.5, \]
\[ \beta / \gamma = 5 / 9 \quad \text{and} \quad \hat{d}_0 = 2 \text{ m} \]
\[ f = 0.03 \text{ m} \]

(fully calibrated camera)

* G.L. Mariottini and D. Prattichizzo, *Robotics and Automation Magazine* ‘05
IBVS pinhole: Simulation Results (II)

Second Step

$\kappa_t = 100$
First Step: partially calibrated camera
\[ \hat{f} = 0.005 \text{ m} \]
(the other parameters are equal to the previous)
**IBVS pinhole: Experimental Results (I)**

Pioneer 3X-DE by ActivMedia

Lumenera LU175C color CCD camera

Connected via RS232 to a Pentium 4, 2GHz, 640 Mb Ram

\[ f = 0.004 \text{ m}, \quad u_0 = 635, \quad v_0 = 512 \text{ pixels} \]

**M-estimator:** robust estimation of the epipoles implemented with Intel’s OpenCV libraries
IBVS pinhole: Experimental Results (III)

Gian Luca Mariottini, Univ. Texas at Arlington
IBVS pinhole: Experimental Results (IV)

Gian Luca Mariottini, Univ. Texas at Arlington