An Introduction to Wheeled Mobile Robot Kinematics and Dynamics

Giovanni Indiveri

University of Lecce - DII,
Department of Innovation Enginnering
Via Monteroni,
73100 Lecce, Italy

giovanni.indiveri@unile.it

RoboCamp - Paderborn (Germany) April 8, 2002

Outline

Kinematics:

A branch of dynamics that deals with aspects of motion apart from considerations of mass and force (Merriam-Webster Dictionary)

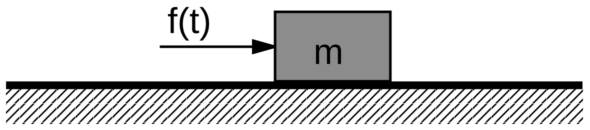
Branch of physics and a subdivision of classical mechanics concerned with the geometrically possible motion of a body or system of bodies without consideration of the forces involved (Encyclopedia Britannica)

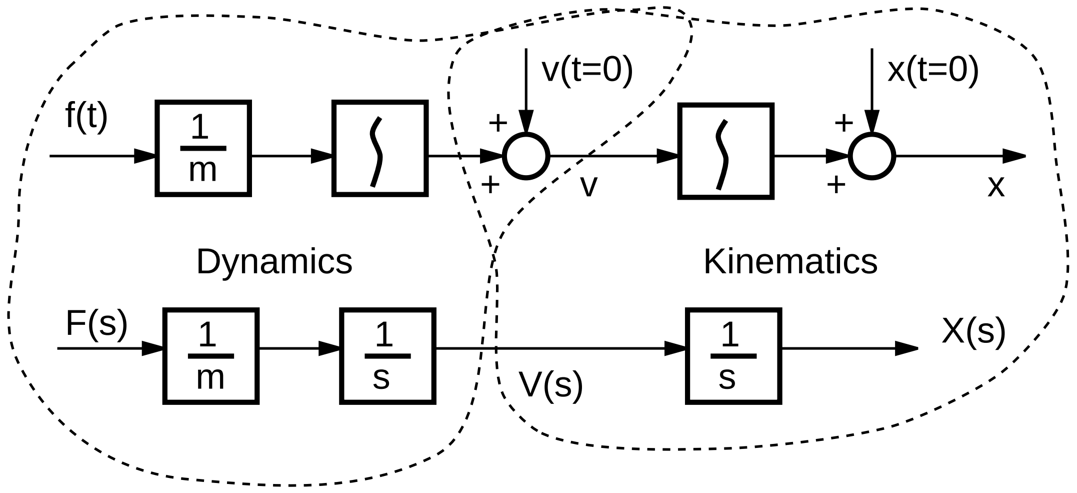
Dynamics:

A branch of mechanics that deals with forces and their relation primarily to the motion but sometimes also to the equilibrium of bodies (Merriam-Webster Dictionary)

Branch of physical science and subdivision of mechanics that is concerned with the motion of material objects in relation to the physical factors that affect them: force, mass, momentum, energy. (Encyclopedia Britannica)

A 1D example





$$f = ma = m\frac{dv}{dt} = m\dot{v}$$

$$v(t) - v_{t_0} = \frac{1}{m}\int_{t_0}^{t} f(\tau)\,d\tau$$

$$\dot{v} = \frac{1}{m}f$$

$$- - - - - - - - - - - -$$

$$v = \frac{dx}{dt}$$

$$x(t) - x_{t_0} = \int_{t_0}^{t} v(\tau)\,d\tau$$

$$\dot{x} = v$$

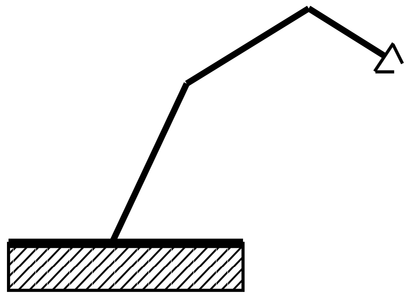
Dynamics versus Kinematics

Dynamics always depends on the systems kinematics.

Assuming the velocity to be the system input, the kinematics is a purely differential - geometrical issue independent from the dynamics.

Eg. Robotic manipulator: $v = J(q)\dot{q}$ and $a = \dot{J}(q)\dot{q} + J(q)\ddot{q}$

$$M(q)\ddot{q} + C(q,\dot{q}) + (F(q,\dot{q})) = \tau$$
$$v = J(q)\dot{q}$$

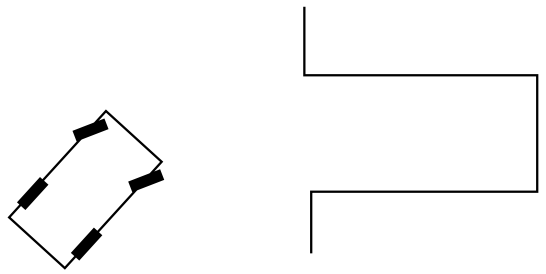


Mobile Robots

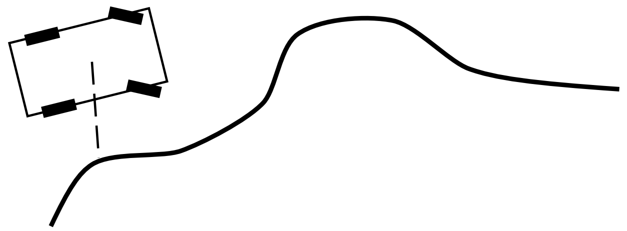
∗ Tracks
∗ Wheels (traditional or omnidirectional)
∗ Legs
∗ Hybrid
∗ Other (eg. hovercraft)

Motion Control Tasks

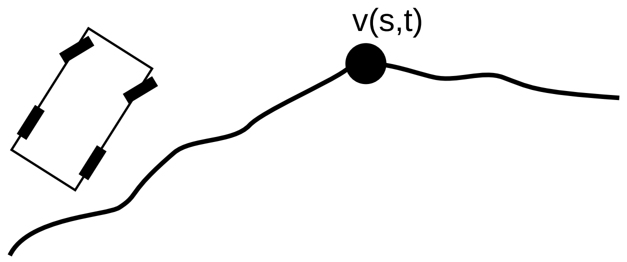
∗ Pose regulation (parking)



∗ Path following
(requires planning)

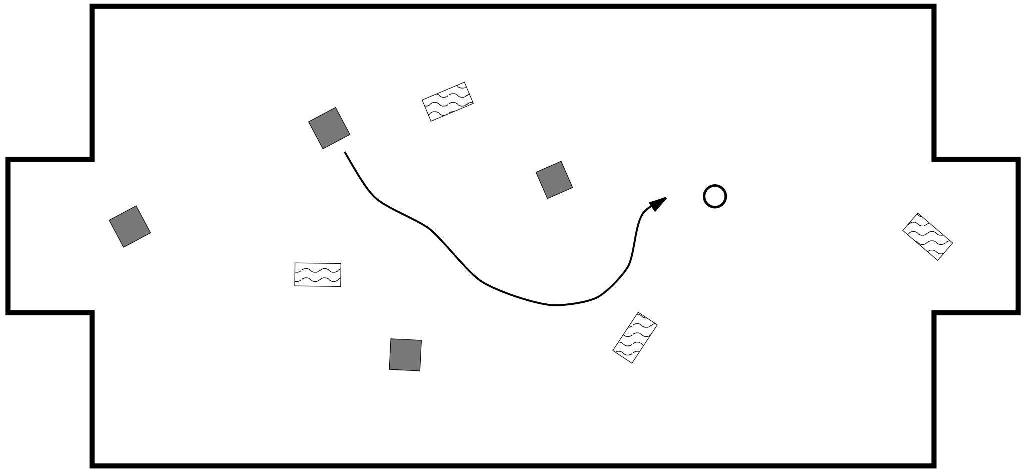


∗ Trajectory tracking
(requires planning)



RoboCup Scenario

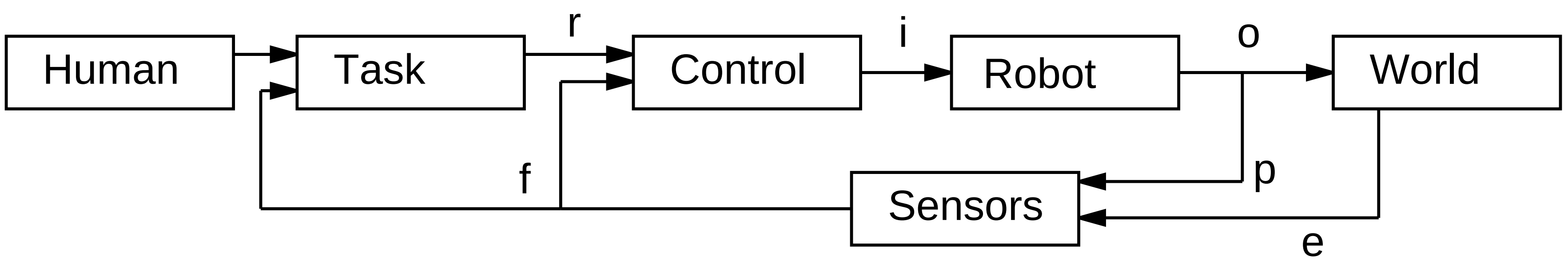
$$\left.\begin{array}{l}\text{Scructured}\\ \text{Dynamical}\\ \text{Complex Task}\end{array}\right\} \Longrightarrow \quad \text{Mix of} \left\{\begin{array}{l}\text{Pose regulation}\\ \text{Path following}\\ \text{Trajectiry tracking}\\ \text{Local reactive behaviour}\end{array}\right.$$



Reactive behaviour (eg.)

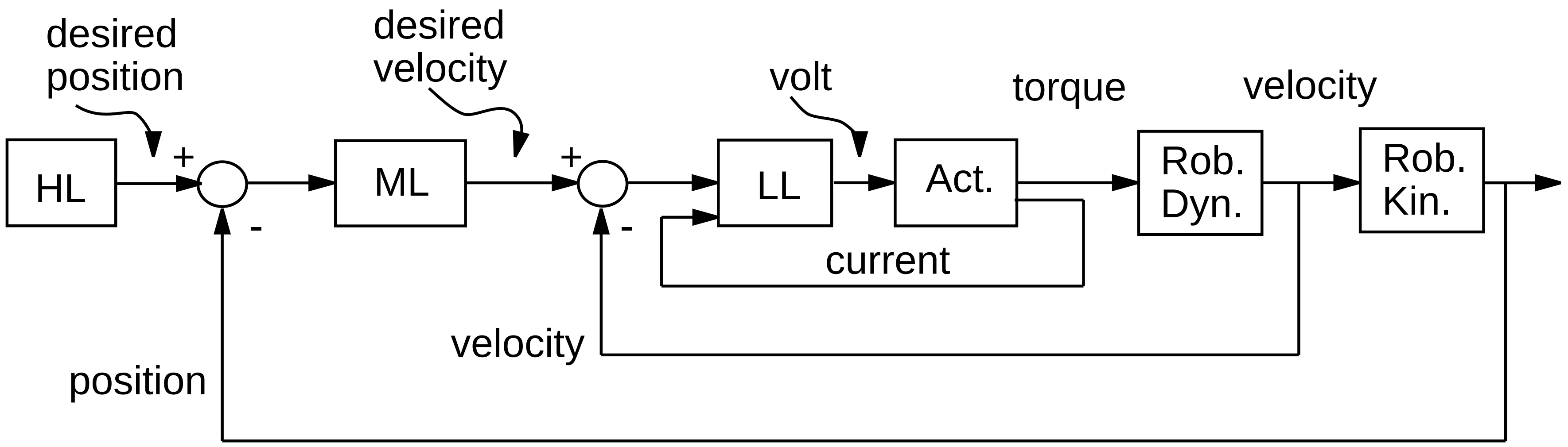
* Obstacle avoidance
* Line following
* Wall following
* Target (eg. ball) following
* Kicking
* etc.

Robot Functional Block Diagrams



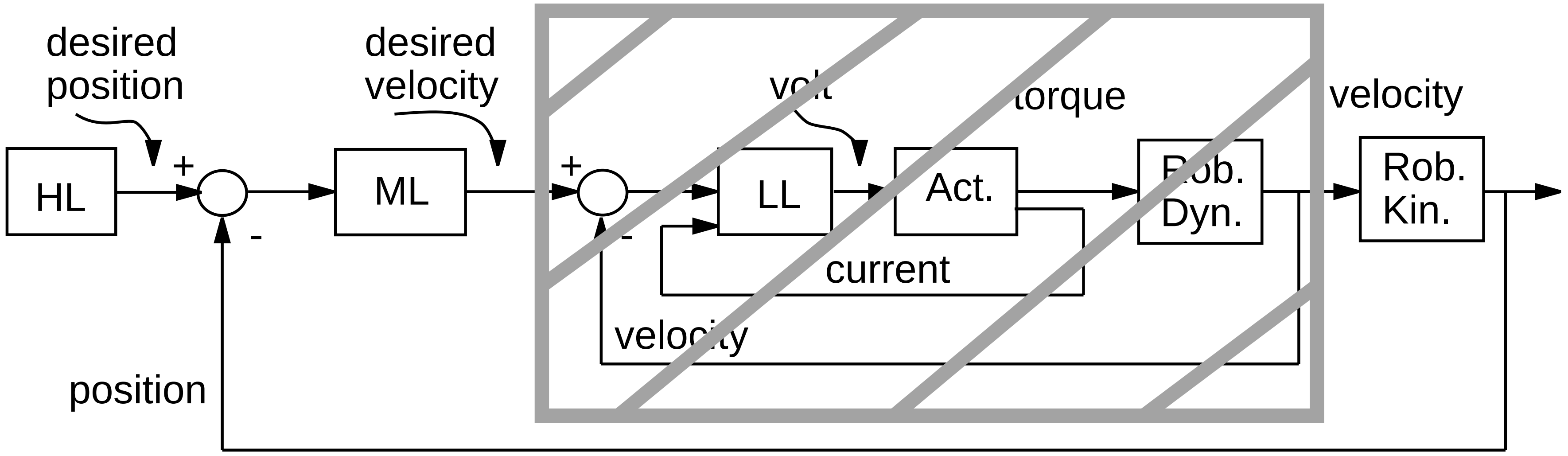
- $*$ r = references
- $*$ i = inputs
- $*$ o = outputs (effectors and motion actuators)
- $*$ f = feedback
- $*$ p = proprioceptive (encoders, gyros, etc.)
- $*$ e = exteroceptive (vision, bumbers, laser, etc.)

Motion Control Functional Block Diagram



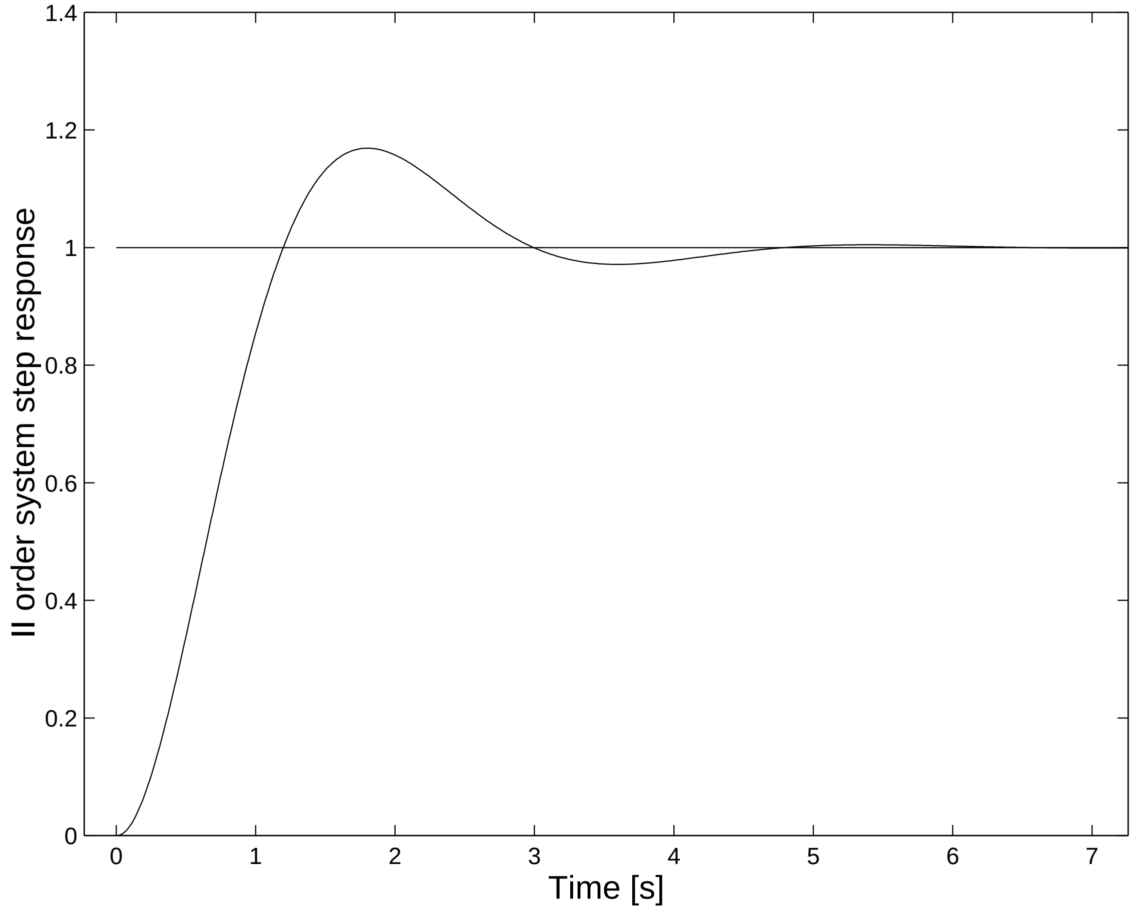
* HL = high level control
* ML = medium level control
* LL = low level control

Kinematic Control

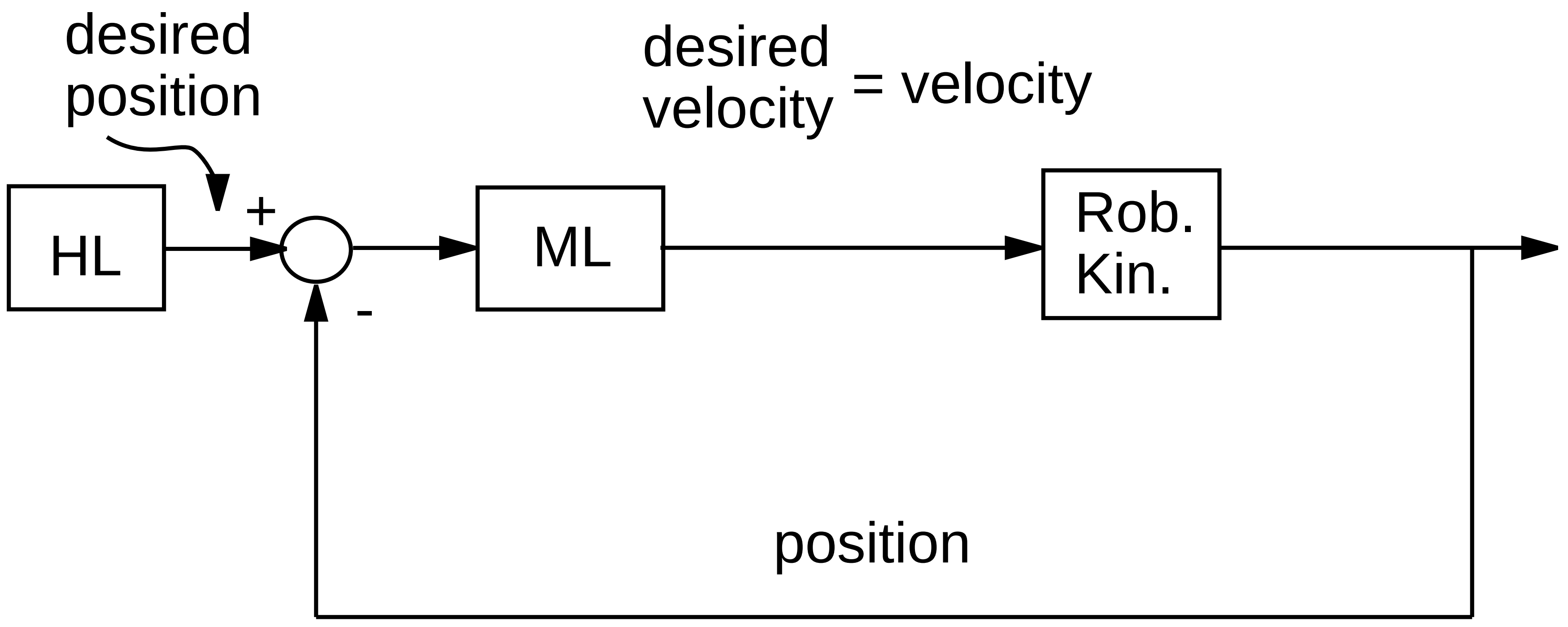


Hp.: desired velocity ≡ real velocity

Kinematic Control, beware of delays



Kinematic Control



Kinematic & Dynamic Models

$$v = J(q)\dot{q}$$
$$M(q)\ddot{q} + C(q,\dot{q}) + (F(q,\dot{q})) = \tau$$

$q =$ generalized coordinates $=$
$+\#$ of coordinates of all the interconnected bodies
$-\#$ number of holonomic constraints.

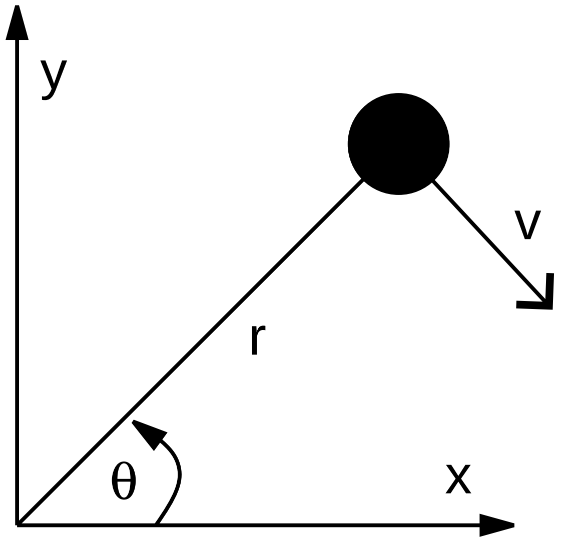
$dim(q) = \#$ degrees of freedom

Holonomic versus Nonholonomic constraints

$*$ Holonomic constraint $h(q) = 0 \Rightarrow$ geometric constraint, i.e. reduction of the degrees of freedom.

$*$ Holonomic constraint $h(q) = 0 \Rightarrow \frac{\partial h}{\partial q}(q)\dot{q} = 0$ an *integrable* kinematic constraint.

$$
\begin{aligned}
&\dot{x} = v_x \\
&\dot{y} = v_y \\
&q \equiv (x, y)^T \\
&\|q\|^2 = x^2 + y^2 = r^2 \ \text{ hol. cons.} \Rightarrow \\
&2x\dot{x} + 2y\dot{y} = 0 \Rightarrow \\
&q^T\dot{q} = 0 \text{ integrable kin. cons.}
\end{aligned}
$$



Example of integrability of a holonomic constraints

$$2x\dot{x} + 2y\dot{y} = 0 \quad \Longrightarrow \quad 2\left(\int_{t_0}^{t} x\dot{x}d\tau + \int_{t_0}^{t} y\dot{y}d\tau\right) = 0 \quad \Longrightarrow$$

$$\int_{x_0}^{x_t} xdx + \int_{y_0}^{y_t} ydy = \frac{1}{2}\left(x_t^2 + y_t^2 - (x_0^2 + y_0^2)\right) = 0 \quad \Longrightarrow$$

$$x(t)^2 + y(t)^2 = const. \quad \Longleftrightarrow \quad x\dot{x} + y\dot{y} = 0$$

Holonomic versus Nonholonomic constraints

$*$ Nonholonomic constraint $h(q,\dot{q}) = 0$ is a kinematic constraint that *cannot* be integrated to yield a geometric constraint.

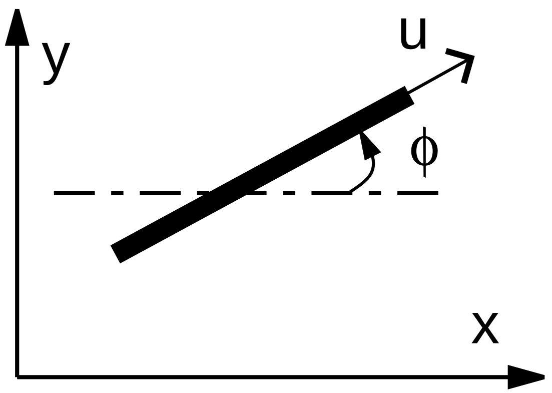
$*$ A (first order) nonholonomic constraint does not reduce the geometrical degrees of freedom.

$$\begin{cases} \dot{x} = u \cos\phi & q \equiv (x, y, \phi)^T \\ \dot{y} = u \sin\phi & v \equiv (u, \omega)^T \\ \dot{\phi} = \omega \end{cases}$$

$$\frac{dy}{dx} = \tan\phi \Rightarrow \dot{y}\cos\phi - \dot{x}\sin\phi = 0$$

$$h(q,\dot{q}) = A(q)\dot{q} =$$

$$= (\sin\phi, -\cos\phi, 0)(\dot{x}, \dot{y}, \dot{\phi})^T = 0$$



Pfaffian Kinematic Constraints

$$A(q)\dot{q} = 0 \begin{cases} \text{integrable} \;\Rightarrow\; \text{holonomic constraint} \\ \text{nonintegrable} \;\Rightarrow\; \text{nonholonomic constraint} \end{cases}$$

If the constraint is holonomic, q must satisfy some equation $h(q) = 0$ and thus the degrees of freedom are reduced.

If the constraint is nonholonomic, q is free and $\dot{q}$ must belong to $Ker(A(q))$, i.e. it is constrained.

Integrability of Kinematic Constraints

Given the scalar constraint $a(q)^T \dot{q} = 0$ is it holonomic or nonholonomic? The answer is by no means trivial ...

Holonomic means that there $\exists\ h(q) : R^n \to R \quad : \quad h(q) = 0 \ \cup \ \dot{h}(q) = \sum_{i=1}^{n} \frac{\partial h}{\partial q_i} \dot{q}_i = 0$ such that

$$\sum_{i=1}^{n} a_i(q)\dot{q}_i = 0 \iff \sum_{i=1}^{n} \frac{\partial h}{\partial q_i} \dot{q}_i = 0$$

In turn this requires that $\exists\ \beta(q) : R^n \to R$ (*integrating factor*) such that

$$\frac{\partial h}{\partial q_j} = \beta(q) a_j(q) \ \forall \ j$$

Integrability of Kinematic Constraints

$$\frac{\partial h}{\partial q_j} = \beta(q) a_j(q) \;\; \forall \;\; j$$

Remembering that for a C^2 function $\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2 f}{\partial y \partial x}$ (Schwartz Theorem)

$$\frac{\partial^2 h}{\partial q_i \partial q_j} = \frac{\partial^2 h}{\partial q_j \partial q_i} \implies \frac{\partial \beta(q)\, a_i(q)}{\partial q_j} = \frac{\partial \beta(q)\, a_j(q)}{\partial q_i} \;\; \forall \;\; i, j = 1, 2, \ldots, n$$

The "test" of integrability can be made checking the existence of the solution of a PDE (differential geometry tools).

Kinematic Models of Wheeled Mobile Robots

$v = J(q)\dot{q} \quad : \quad q \in R^{n\times 1}$

$A(q)\dot{q} = 0 \quad : \quad A(q) \in R^{m\times n}, m < n \cup Rank(A) = m$

$S(q) = [s_1(q)\ s_2(q) \ldots s_{n-m}(q)] \in R^{n\times(n-m)} \quad :$
$\{s_i(q) \in R^{n\times 1} \quad \forall i = 1, 2, \ldots, n-m\}$ is a basis of $\ker(A)$

$\dot{q} = S(q)u$ is by construction a feasible velocity as

$A(q)S(q)u = 0 \quad u \in R^{(n-m)\times 1}$ **pseudovelocity**

Kinematic Models of Wheeled Mobile Robots

As wheeled vehicles are affected by nonholonomic constraints, their kinematics can be always written as

$$\dot{q} = S(q)u$$

with

$$\begin{array}{rcl} q & \in & R^{n\times 1} \ : \ n \ \text{\# of degrees of freedom} \\ u & \in & R^{p\times 1} \ : \ p \ \text{dimension of input vector} \\ p & \leq & n \ : \ \ \text{underactuated system} \end{array}$$

Ideal Unicycle

$$q = (x, y, \phi)^T$$

$$u = (v, \omega)^T$$

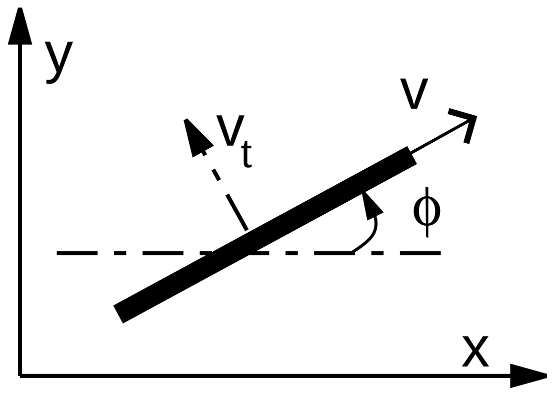
$$\begin{cases} \dot{x} = v \cos\phi \\ \dot{y} = v \sin\phi \\ \dot{\phi} = \omega \end{cases}$$

$$S(q) = \begin{bmatrix} \cos\phi & 0 \\ \sin\phi & 0 \\ 0 & 1 \end{bmatrix}$$

$$\dot{q} = S(q)u$$

$$v_t = 0 \implies \dot{y}\cos\phi - \dot{x}\sin\phi = 0$$

$$A(q) = (-\sin\phi, \cos\phi, 0) \quad : \quad A(q)S(q) = 0$$

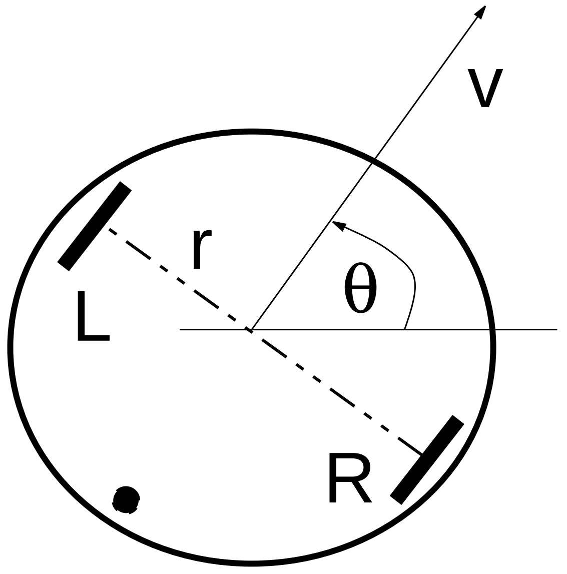


"Pioneer" - Like Kinematics

$$q = (x, y, \theta)^T$$
$$u = (v_R, v_L)^T$$

$$\dot{q} = \begin{bmatrix} \cos\theta/2 & \cos\theta/2 \\ \sin\theta/2 & \sin\theta/2 \\ \frac{1}{2r} & -\frac{1}{2r} \end{bmatrix} u$$

$$\dot{q} = S(q)u$$
$$v_R = r\,\dot{\psi}_R$$
$$v_L = r\,\dot{\psi}_L$$



References

A large number of kinematic models for car-like robots are described in

Robot Motion Planning and Control, edited by J.-P. Laumond, chapter ”Feedback control of a nonholonomic car-like robot”, by A. DeLuca, G. Oriolo, C. Samson, Springer-Verlag 1998, ISBN 3-540-76219-1, downloadable from http://www.laas.fr/ jpl/book.html.

Dynamic Modelling

$L = T - V$ Kinetic - Potential energy

$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \tau$ unconstrained dynamics of a conservative system

Pfaffian constraints $A(q)\dot{q} = 0$

$\exists \ \lambda(q, \dot{q}) : \tau_{cons} = -A^T(q)\lambda$ as $\tau_{cons}^T \dot{q} = 0$

$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \tau - A^T(q)\lambda$ constrained dynamics assuming that τ_{cons} does no work.

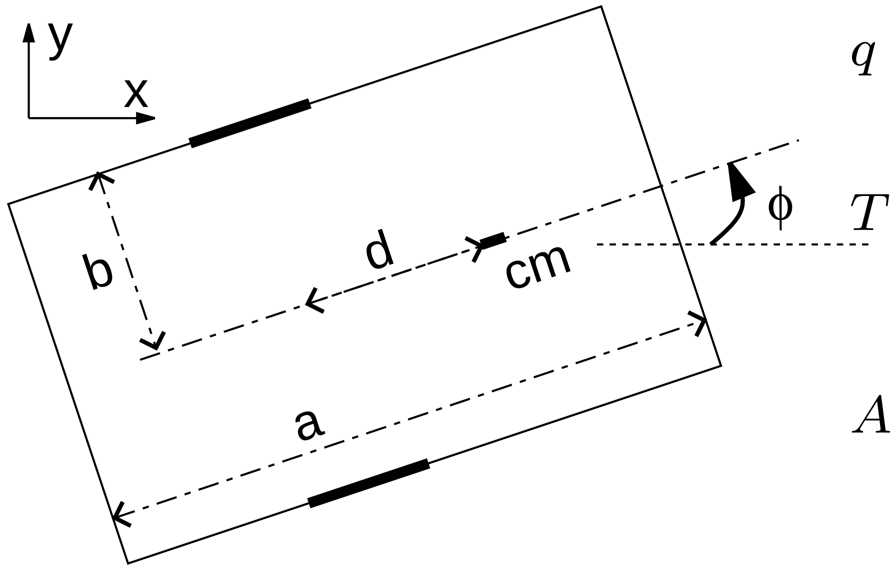
Dynamic Equations

Examples of dynamic modelling can be found in:

"Adaptive Tracking Control of a Nonholonomic Mobile Robot", by T. Fukao, H. Nakagawa, N. Adachi, *IEEE Trans. on Robotics and Automation*, vol. 16, no. 5, October 2000, pp. 609-615.

"Control of Mechanical Systems with Rolling Constraints: Application to Dynamic Control of Mobile Robots", by N. Sarkar, X. Yun, V. Kumar, *Int. Journal of Robotics Research*, vol. 13, no.1, 1994, pp. 55-69.

Dynamic Equations



$$q = (x_c, y_c, \phi, \theta_r, \theta_l)^T$$

$$T = \frac{1}{2} m_c v_c^2 + \frac{1}{2} I_c \dot{\phi}^2 + T_m$$

$$A(q) = \begin{bmatrix} -\sin\phi & \cos\phi & -d & 0 & 0 \\ -\cos\phi & -\sin\phi & -b & r & 0 \\ -\cos\phi & -\sin\phi & b & 0 & r \end{bmatrix}$$

$$S(q) : A(q)S(q) = 0$$

Dynamic Equations

$$M(q)\ddot{q} + V(q,\dot{q}) = B(q)\tau - A^T(q)\lambda$$

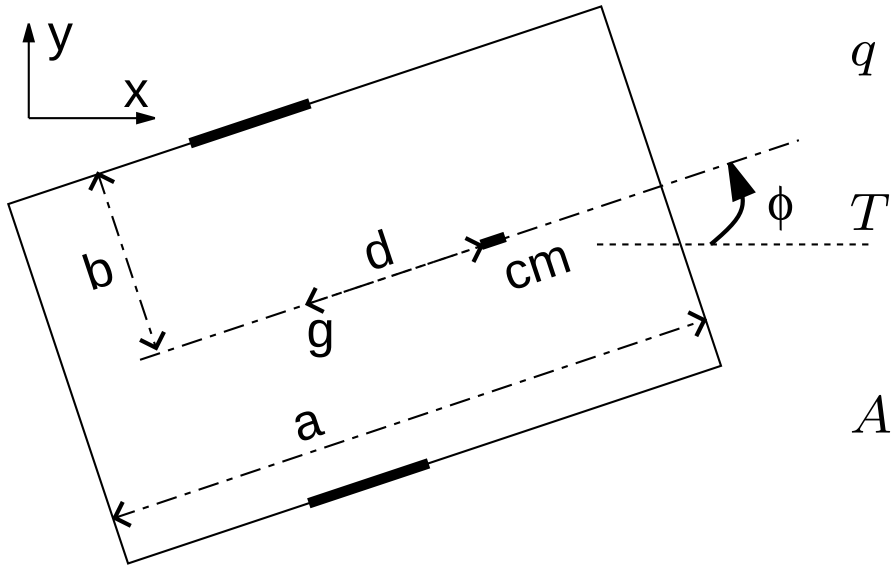
$$M(q) = \begin{bmatrix} m & 0 & 2m_w d\sin\phi & 0 & 0 \\ 0 & m & -2m_w d\cos\phi & 0 & 0 \\ 2m_w d\sin\phi & -2m_w d\cos\phi & I & 0 & 0 \\ 0 & 0 & 0 & I_w & 0 \\ 0 & 0 & 0 & 0 & I_w \end{bmatrix}$$

$$I = I_c + 2m_w(d^2 + b^2) + 2I_m \quad ; \quad m = m_c + 2m_w$$

Dynamic Equations

$$V(q,\dot{q}) = \begin{bmatrix} 2m_w d\dot{\phi}^2 \cos\phi \\ 2m_w d\dot{\phi}^2 \sin\phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad ; \quad B(q) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad ; \quad \tau = \begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix}$$

Dynamic Equations



$$q = (x_g, y_g, \phi, \theta_r, \theta_l)^T$$

$$T = \frac{1}{2} m_c v_c^2 + \frac{1}{2} I_c \dot{\phi}^2 + T_m$$

$$A(q) = \begin{bmatrix} \sin\phi & -\cos\phi & 0 & 0 & 0 \\ \cos\phi & \sin\phi & b & -r & 0 \\ \cos\phi & \sin\phi & -b & 0 & -r \end{bmatrix}$$

$$S(q) : A(q)S(q) = 0$$

Dynamic Equations

$$M(q)\ddot{q} + V(q,\dot{q}) = B(q)\tau - A^T(q)\lambda$$

$$M(q) = \begin{bmatrix} m & 0 & -m_c d \sin\phi & 0 & 0 \\ 0 & m & m_c d \cos\phi & 0 & 0 \\ -m_c d \sin\phi & m_c d \cos\phi & I & 0 & 0 \\ 0 & 0 & 0 & I_w & 0 \\ 0 & 0 & 0 & 0 & I_w \end{bmatrix}$$

$$I = I_c + 2m_w(d^2 + b^2) + 2I_m \quad ; \quad m = m_c + 2m_w$$

Dynamic Equations

$$V(q,\dot{q}) = \begin{bmatrix} -m_c\, d\, \dot{\phi}^2 \cos\phi \\ -m_c\, d\, \dot{\phi}^2 \sin\phi \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$B(q) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad ; \quad \tau = \begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix}$$

Dynamic Equations

$$q = (x_g, y_g, \phi, \theta_r, \theta_l)^T \ : \ A(q) = \begin{bmatrix} \sin\phi & -\cos\phi & 0 & 0 & 0 \\ \cos\phi & \sin\phi & b & -r & 0 \\ \cos\phi & \sin\phi & -b & 0 & -r \end{bmatrix}$$

$$S(q) \ : \ A(q)S(q) = 0 \quad \Longrightarrow \quad S(q) = \begin{bmatrix} \frac{r}{2}\cos\phi & \frac{r}{2}\cos\phi \\ \frac{r}{2}\sin\phi & \frac{r}{2}\sin\phi \\ \frac{r}{2b} & -\frac{r}{2b} \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\dot{q} = S(q)u \ : \ u = (\dot{\theta}_r, \dot{\theta}_l)^T$$

$$M(q)\ddot{q} + V(q, \dot{q}) = B(q)\tau - A^T(q)\lambda$$

$$\ddot{q} = \dot{S}(q)\,u + S(q)\dot{u} \quad \Longrightarrow$$

$$M(q)S(q)\dot{u} + \Big(V(q, \dot{q}) + M(q)\dot{S}(q)\,u\Big) = B(q)\tau - A^T(q)\lambda$$

Dynamic Equations

$$M(q)S(q)\dot{u} + \Big(V(q,\dot{q}) + M(q)\dot{S}(q)\,u\Big) = B(q)\tau - A^T(q)\lambda \quad \Longrightarrow$$

$$S^T(q)M(q)S(q)\dot{u} + \Big(S^T(q)V(q,\dot{q}) + S^T(q)M(q)\dot{S}(q)\,u\Big) =$$
$$= S^T(q)B(q)\tau - S^T(q)A^T(q)\lambda$$

$$S^T(q)A^T(q)\lambda = 0 \qquad q = (x_g, y_g, \phi, \theta_r, \theta_l)^T \;\; ; \;\; u = (\dot{\theta}_r, \dot{\theta}_l)^T$$

$$\bar{M}(q)\dot{u} + \bar{V}(q,u) = \bar{B}(q)\tau \quad \begin{cases} \dot{q} = S(q)\,u \\ \dot{u} = -\bar{M}^{-1}(q)\bar{V}(q,u) + \bar{M}^{-1}(q)\bar{B}(q)\tau \end{cases}$$

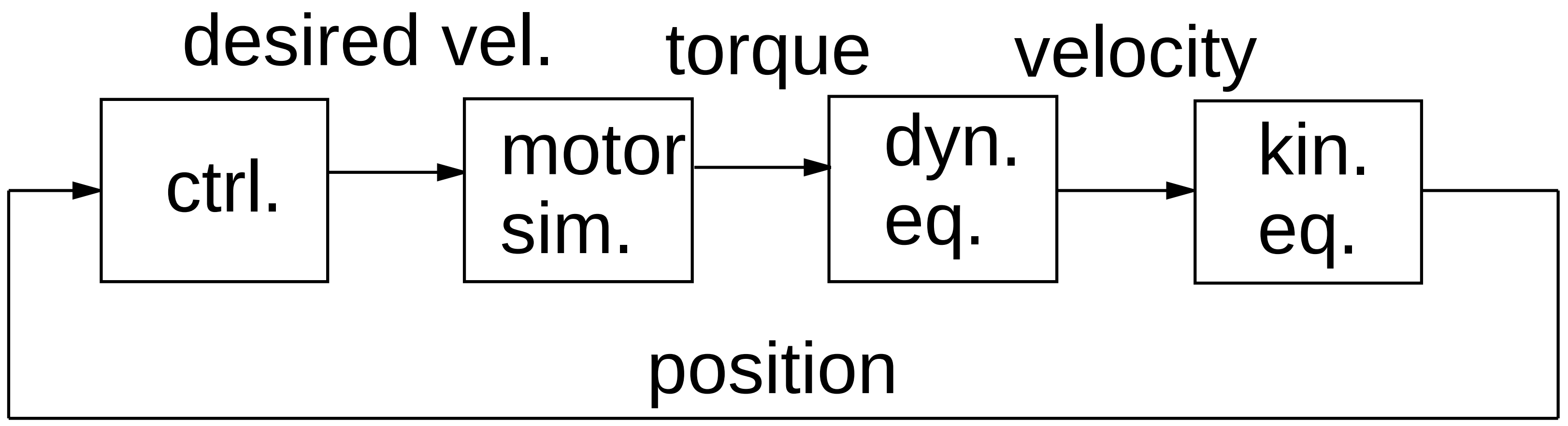
Modelling Applications

Why bother with all this modelling ?

- ∗ Analysis
- ∗ Simulation
- ∗ Control

Simulation

Simulators can be purely kinematic or dynamic. Dynamic simulators require the knowledge of the dynamic equation parameters.



Simulation

$$\begin{cases} \dot{q} = S(q)\,u \\ \dot{u} = -\bar{M}^{-1}(q)\bar{V}(q,u) + \bar{M}^{-1}(q)\bar{B}(q)\tau \end{cases}$$

In order to implement the simulation equations the geometrical and inertial parameters of the dynamic equations must be known.

Linear system identification techniques, eg. least squares algorithms, may be employed.

Control

$$\begin{cases} \dot{q} = S(q)\,u & \text{System K} \\ \dot{u} = -\bar{M}^{-1}(q)\bar{V}(q,u) + \bar{M}^{-1}(q)\bar{B}(q)\tau & \text{System D} \end{cases}$$

In state space notation $x = (q^T, u^T)^T \quad \Longrightarrow$

$$\dot{x} = \begin{bmatrix} S(q)\,u \\ -\bar{M}^{-1}(q)\bar{V}(q,u) \end{bmatrix} + \begin{bmatrix} 0 \\ \bar{M}^{-1}(q)\bar{B}(q) \end{bmatrix} \tau$$

$$\dot{x} = f(x) + g(x)\tau \qquad \text{Nonlinear control system with drift}$$

Is the system controllable ?

Controllability

Given $\dot{x} = f(x) + g(x)\tau : x \in \mathcal{M}$ and $\tau \in \mathcal{U}$, the class of piecewise continuous time functions, call $x(t, 0, x_0, \tau)$ the solution of the state equation.

The system is said to be *controllable* if

$$\forall\, x_1, x_2 \in \mathcal{M},\ \ \exists\, T < \infty,\ \ \exists\, \tau : [0, T] \rightarrow \mathcal{U}\ :\ x(T, 0, x_1, u) = x_2.$$

For a linear time invariant system $\dot{x} = Ax + B\tau$ this can be checked by a simple geometrical condition (rank of the Kalman controllability matrix).

For nonlinear systems, the test is more complex and based on differential geometry tools.

Controllability

Notice that the notion of controllability refers to the existence of *open-loop* controls (input signals) able to steer the state from any initial state to any final one.

The controllability issue of the dynamic model of a wheeled mobile robot can be addressed on the basis of the following two results.

Controllability

$$\begin{cases} A(q)\dot{q} \text{ is completely nonholonomic} \\ \\ \Longleftrightarrow \\ \\ \dot{q} = S(q)\,u \text{ is controllable} \end{cases}$$

$$\begin{cases} \dot{q} = S(q)\,u \text{ is controllable} \\ \\ \Longleftrightarrow \\ \\ \left.\begin{array}{l} \dot{q} = S(q)\,u \\ \dot{u} = \tilde{\tau} \end{array}\right\} \text{ is controllable} \end{cases}$$

Feedback Linearization & Dynamic Extension

$$\begin{cases} \dot{q} = S(q)\, u & \text{System K} \\ \dot{u} = -\bar{M}^{-1}(q)\bar{V}(q,u) + \bar{M}^{-1}(q)\bar{B}(q)\tau & \text{System D} \end{cases}$$

If dim(τ) $\geq$ dim(u) and the necessary (pseudo)inverses exist, choose the feedback linearizing control input

$$\tau = \left(\bar{M}^{-1}(q)\bar{B}(q)\right)^{\#} \left(+\bar{M}^{-1}(q)\bar{V}(q,u) + \tilde{\tau}\right) \quad \Longrightarrow$$

$$\begin{cases} \dot{q} = S(q)\, u \\ \dot{u} = \tilde{\tau} \end{cases} \quad \text{dynamic extension of purely kinematic model.}$$

Consequences

$*$ A completely nonholonomic kinematic model is always controllable, thus the *path planning* problem is always well posed: "Path Planning and Navigation in RoboCup", Pedro Lima, RoboCamp 2002.

$*$ The control problem is "basically" a kinematic issue.

References

”Control of Nonholonomic Systems”, by Giuseppe Oriolo. Slides of a PhD course held at La Sapienza University of Rome, April-May 2001. http://labrob.ing.uniroma1.it/people/oriolo/cns/cns.htm

”A Mathematical Introduction to Robotic Manipulation”, by R.M.Murray, Z.Li, and S.Sastry, CRC Press, 1994.

Control problem formulation

What is control actually about?

$$\dot{x} = f(x, u)$$

Intuitively, find $u(t) : x(t) \to \bar{x}$ and "stays" there, i.e. make $\bar{x}$ an equilibrium point for f. This means that the minimal requirement is that $f(\bar{x}, u(\bar{x}, \cdot)) = 0$

$$z = x - \bar{x} \quad \Longrightarrow \quad \dot{z} = f(x, u) - f(\bar{x}, u(\bar{x}, \cdot)) = f(z + \bar{x}, u(z + \bar{x}, \cdot))$$

By proper state transformation, driving x to $\bar{x}$ can always be formulated as driving z to zero.

Asymptotic convergence to an equilibrium point

$$\dot{x} = f(x, u)$$

Driving x to zero is a "regulation" problem and can be achieved in open loop.

$$\begin{aligned} \dot{x} &= a\,u \quad : \quad x(t)|_{t=0} = x_0 \\ x(t) &= x_0 e^{-\gamma\, t} \qquad \Longrightarrow \\ \\ u(t) &= -\frac{x_0}{a}\gamma\, e^{-\gamma\, t} \text{ open loop regulating control} \end{aligned}$$

If the knowledge of a should not be perfect, i.e. $\widehat{a} = a \pm \epsilon : \epsilon \neq 0$, the open loop control would fail ...

Robustness to model uncertainty

$$\begin{aligned}
\dot{x} &= a\,u \quad : \quad x(t)|_{t=0} = x_0 \\
u(t) &= -\frac{x_0}{\widehat{a}}\,\gamma\,e^{-\gamma\,t} \qquad \Longrightarrow \\
x(t) &= \frac{a}{\widehat{a}} x_0\,e^{-\gamma\,t} + x_0\left(1 - \frac{a}{\widehat{a}}\right)
\end{aligned}$$

Robustness to external disturbances

$$\begin{aligned}
\dot{x} &= a\,u + \delta \;:\; x(t)|_{t=0} = x_0,\ \delta = const. \\
u(t) &= -\frac{x_0}{a}\gamma\, e^{-\gamma\, t} \qquad \Longrightarrow \\
x(t) &= x_0\, e^{-\gamma\, t} + \delta\, t
\end{aligned}$$

Robustness and stabilization

Feedback is used to achieve robustness by "stabilizing" the equilibrium.

$\dot{x} = f(x,t)$; $x(t_0) = x_0$ then the equilibrium $x^* = 0$
is said to be stable at t_0 in the sense of Lyapunov if
$\forall\, \varepsilon > 0 \;\; \exists\, \delta(\varepsilon, t_0)$ such that
$\|x_0\| < \delta \quad \Longrightarrow \quad \|x(t)\| < \varepsilon \,\forall\, t \geq t_0$

Stabilizing an equilibrium is equivalent to assuring the existence of a non-increasing "energy" function having its minimum in the desired equilibrium.

Stabilization for linear systems

$$\dot{x} = a\,u + \delta \quad : \quad x(t)|_{t=0} = x_0,\ \delta = const.$$

$$u(t) = -k\,x(t) \quad \Longrightarrow \quad \dot{x} = -a\,k\,x(t) + \delta$$

$$x(t) = x_0\,e^{-k\,a\,t} + \frac{\delta}{k\,a}\left(1 - e^{-k\,a\,t}\right)$$

$V = \frac{1}{2}x^2 \;\rightarrow$ Lyapunov function for the nonperturbed system.

Stabizability

Prior to looking for a continuous time invariant stabilizing feedback law, the question is ”does it exist?”

For linear systems, controllability $\Longrightarrow$ stabizability.

Not so for nonlinear systems, in particular:

Nonholonomic mechanical systems, either in kinematic or dynamic form, cannot be stabilized at a point by smooth feedback (R. Brockett, 1983).

The alternatives are time-varying or discontinuous stabilizing feedback laws.

References

Besides the previously mentioned,

”Nonlinear Systems”, by H.K. Khalil, Prentice-Hall, Inc. New Jersey, USA, second edition, 1996.

”Applied Nonlinear Control”, by J.-J. E. Slotine, and W. Li, Prentice-Hall, Inc. New Jersey, USA, 1991.

Trajectory Tracking

Consider the unicycle model. The task is to asymptotically converge on the pose of a reference "target" unicycle:

$$\begin{cases} \dot{x}_r = v_r \cos\phi_r \\ \dot{y}_r = v_r \sin\phi_r \\ \dot{\phi}_r = \omega_r \end{cases} \quad : \quad \omega_r(t) \neq 0 \text{ or } v_r(t) \neq 0 \text{ when } t \to \infty$$

Define the error vector $e = (e_1, e_2, e_3)^T$ as

$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r - x \\ y_r - y \\ \omega_r - \omega \end{pmatrix}$$

i.e. the pose error projected on the unicycle-fixed local frame.

Trajectory Tracking

By differentiating the above, and introducing the new inputs

$$\begin{cases} u_1 = -v + v_r \cos e_3 \\ u_2 = \omega_r - \omega \end{cases}$$ the error dynamics results in:

$$\dot{e} = \begin{pmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} e + \begin{pmatrix} 0 \\ v_r \sin e_3 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

By introducing the candidate Lyapunov function

$$V(e) = \frac{k}{2}\left(e_1^2 + e_2^2\right) + \frac{e_3^2}{2} \quad : \quad k > 0$$

Trajectory Tracking

and imposing its time derivative to be non-increasing, it can be proven that the nonlinear feedback control law:

$$\begin{cases} u_1 = -f_1(v_r, \omega_r)\, e_1 \\ u_2 = -k\, v_r \frac{\sin e_3}{e_3}\, e_2 - f_2(v_r, \omega_r)\, e_3 \\ \forall\ f_1(\cdot,\cdot),\ f_2(\cdot,\cdot) > 0 \text{ on } R \times R - (0,0) \end{cases}$$

guarantees $e = 0$ to be a globally asymptotically stable equilibrium.

References

”Recent Trends in Mobile Robots”, ed. Y.F. Zheng, ch.5 ”Nonlinear Control Design for Mobile Robots”, by C. Canudas de Wit, H. Khennouf, C. Samson, O. J. Sordalen, World Scientific Publishing Co. ISBN 981-02-1511-8, 1993.

In this reference a similar approach is also described for path following and a time-varying feedback solution to the unicycle point stabilization problem is presented.

Applications

Within the RoboCup scenario this kind of approach can be used in several circumstances.

Eg. the goal keeper.

Stabilize the goal keeper on a semi-circle centered in the own goal center and "follow" the ball.

Comments

The above described trajectory tracking controller assumes that the error vector

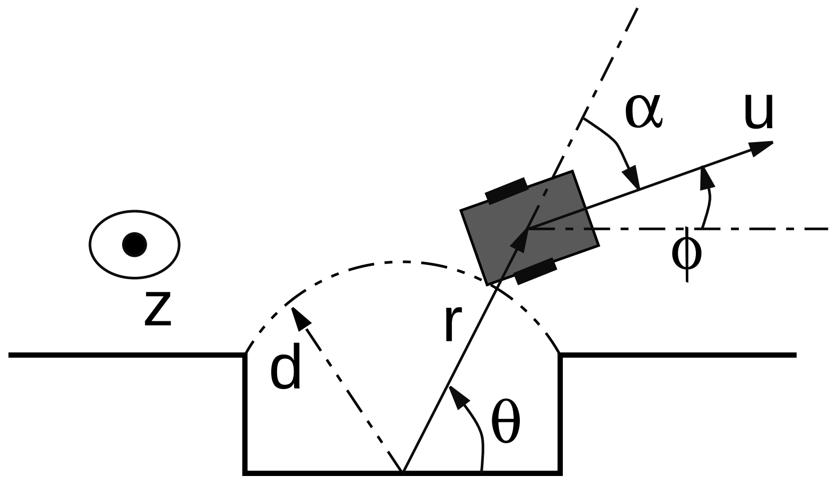
$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r - x \\ y_r - y \\ \omega_r - \omega \end{pmatrix}$$

is available, i.e that the self localization issue is solved.

Moreover planning is needed to generate a suitable reference target unicycle.

Are there alternatives?

Example: goal keeper motion control



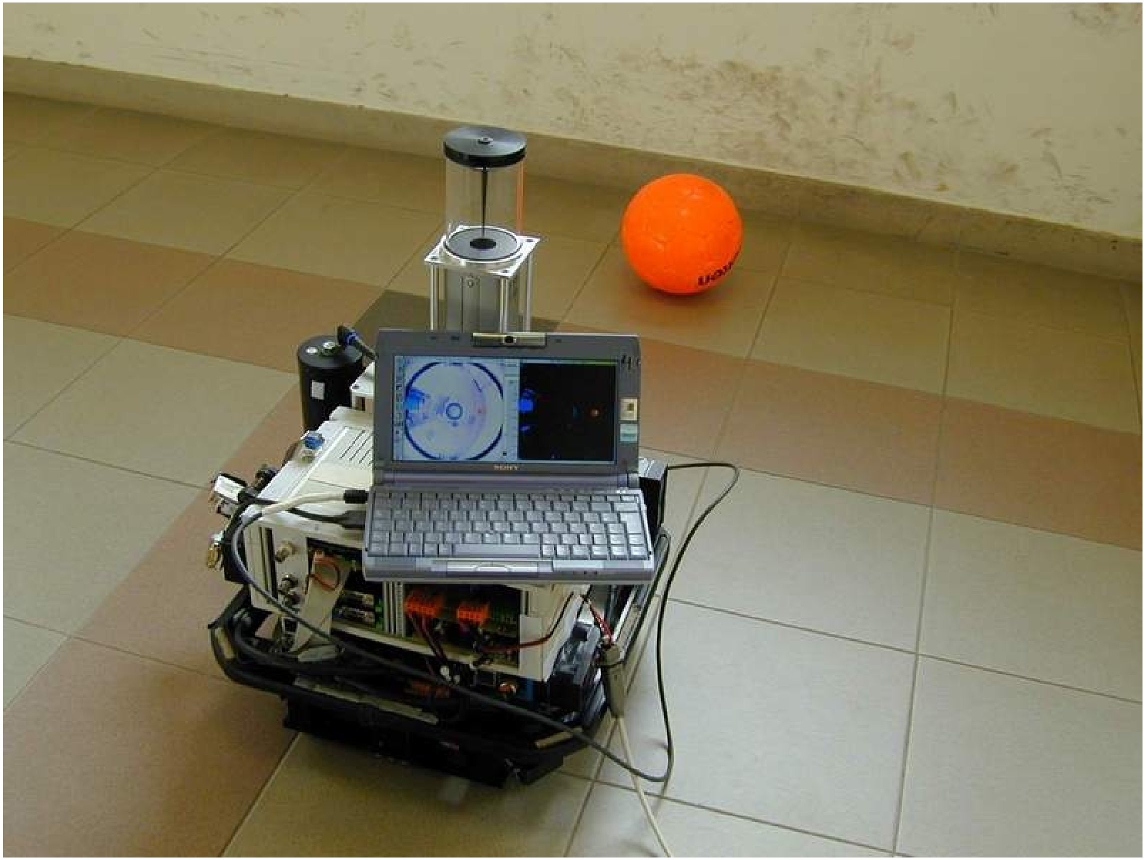
$$\begin{cases} \dot{x} = u \cos\phi \\ \dot{y} = u \sin\phi \\ \dot{\phi} = \omega \end{cases}$$

Stabilize the pose on a circle, stop in front of the ball.

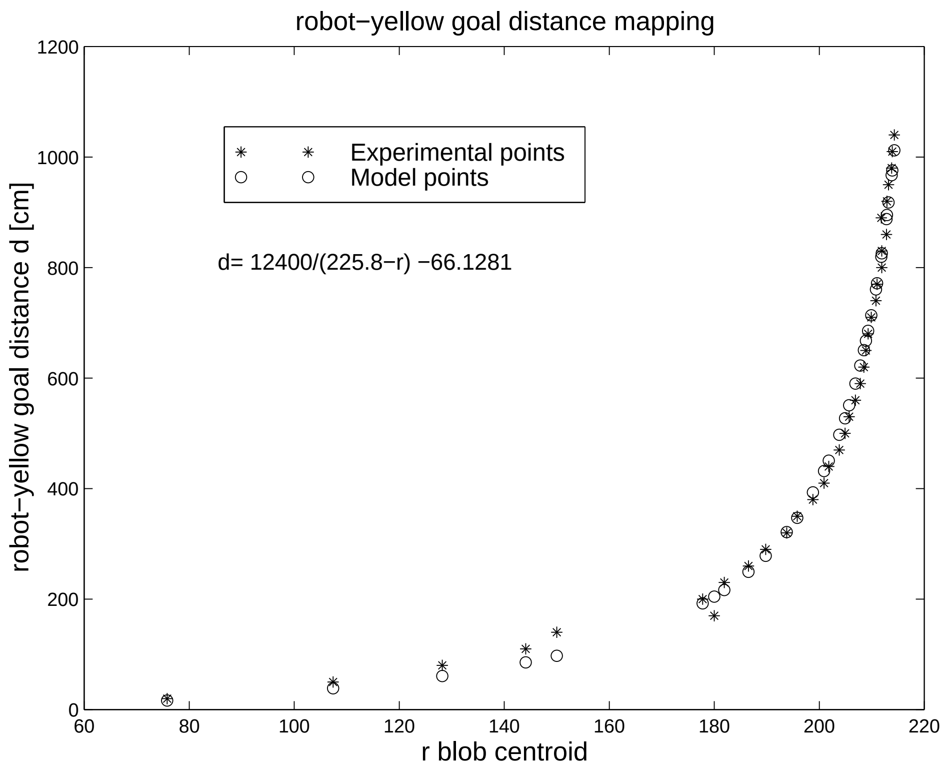
Is trajectory planning/tracking required?

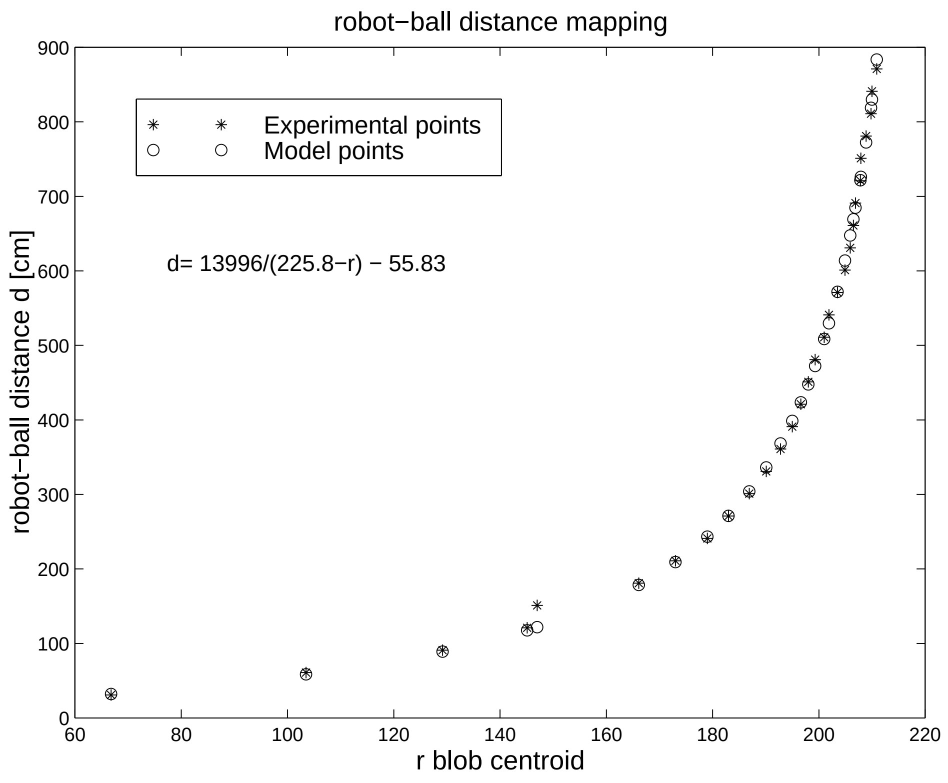
Omnidirectional vision systems allow to measure the goal position *relative* to the robot, i.e. r, α

and similarly the ball position relative to the robot..

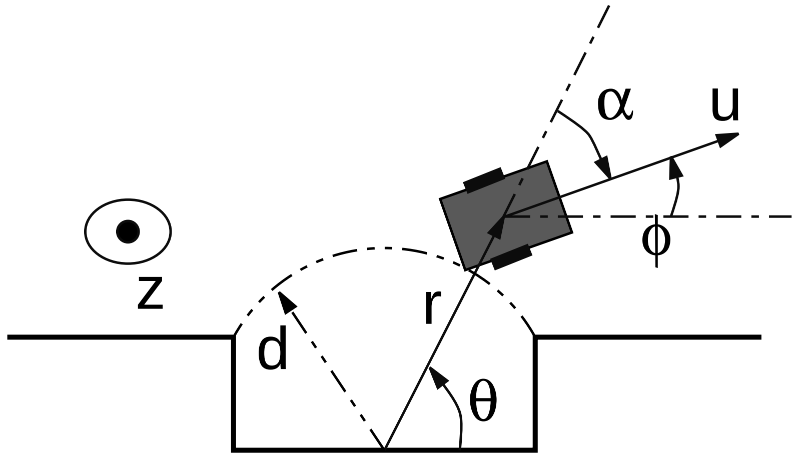


Goal-robot and ball-robot relative position estimation





Nonlinear control design



$$\begin{cases} \dot{x} = u\cos\phi \\ \dot{y} = u\sin\phi \\ \dot{\phi} = \omega \end{cases} \Longrightarrow \begin{cases} \dot{r} = u\cos\alpha \\ \dot{\alpha} = \omega - \frac{u}{r}\sin\alpha \\ \dot{\theta} = \frac{u}{r}\sin\alpha \end{cases}$$

$$V = \frac{1}{2}\left(h\,(d-r)^2 + \left(\frac{\pi}{2}-\alpha\right)^2\right);\, h > 0$$

$$\begin{aligned} \dot{V} &= -h\,(d-r)\dot{r} - \left(\frac{\pi}{2}-\alpha\right)\dot{\alpha} = \\ &= -h\,(d-r)\,u\cos\alpha - \left(\frac{\pi}{2}-\alpha\right)\left(\omega - \frac{u}{r}\sin\alpha\right) \end{aligned}$$

Choose ω to make $\dot{V}$ negative and ...

$$\begin{aligned} \dot{V} &= -h\,(d-r)\,u\cos\alpha - \left(\frac{\pi}{2}-\alpha\right)\left(\omega - \frac{u}{r}\sin\alpha\right) \\ \omega &= \frac{u}{r}\sin\alpha + \gamma\left(\frac{\pi}{2}-\alpha\right) - h\,u\,(d-r)\,\frac{\cos\alpha}{\frac{\pi}{2}-\alpha} \ : \ \gamma, h > 0 \\ &\Longrightarrow \ \dot{V} = -\gamma\left(\frac{\pi}{2}-\alpha\right)^2 \leq 0 \text{ negative semi-definite} \\ V &\geq 0 \ \cup \ \dot{V} \leq 0 \Longrightarrow V \text{ tends to a finite positive limit} \end{aligned}$$

$\dot{V}$ uniformly continuous $\Longrightarrow \dot{V} \to 0$ Barbalats Lemma $\Longrightarrow \alpha \to \dfrac{\pi}{2}$

Asymptotic stability proof

$\alpha \to \frac{\pi}{2} \;\cup\; \dot{\alpha}$ uniformly continuous $\Longrightarrow$

$\dot{\alpha} \to 0$ (Barbalats Lemma) $\Longrightarrow \omega \to \frac{u}{r}$

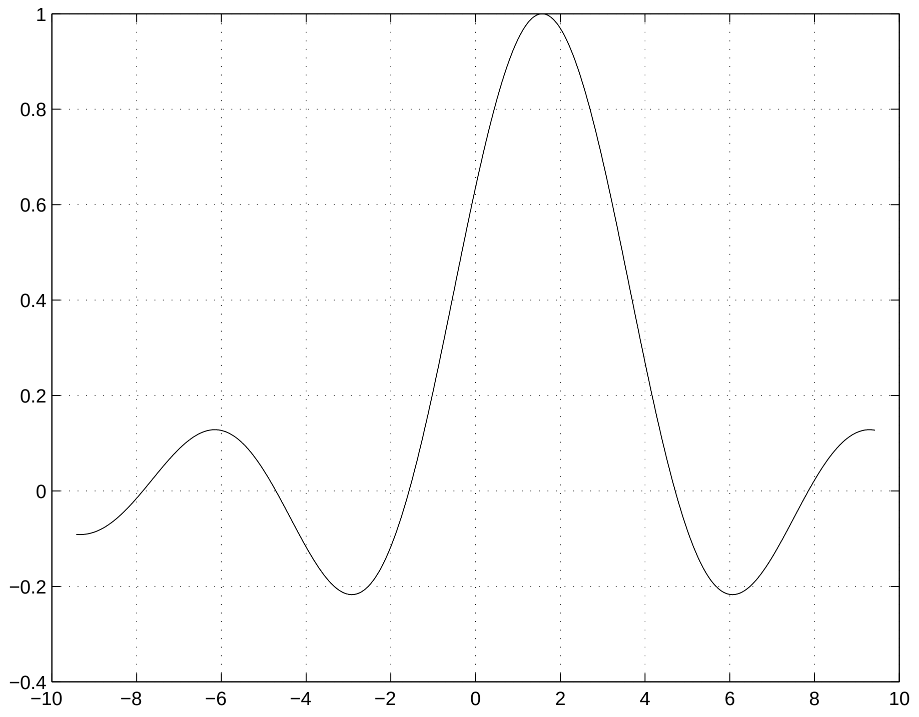
$$\omega = \frac{u}{r}\sin\alpha + \gamma\left(\frac{\pi}{2} - \alpha\right) - h\,u\,(d-r)\,\frac{\cos\alpha}{\frac{\pi}{2}-\alpha} \;:\; \gamma, h > 0$$

$$\lim_{\alpha\to\pi/2} w(t) = \frac{u}{r} + h\,u\,(d-r)$$

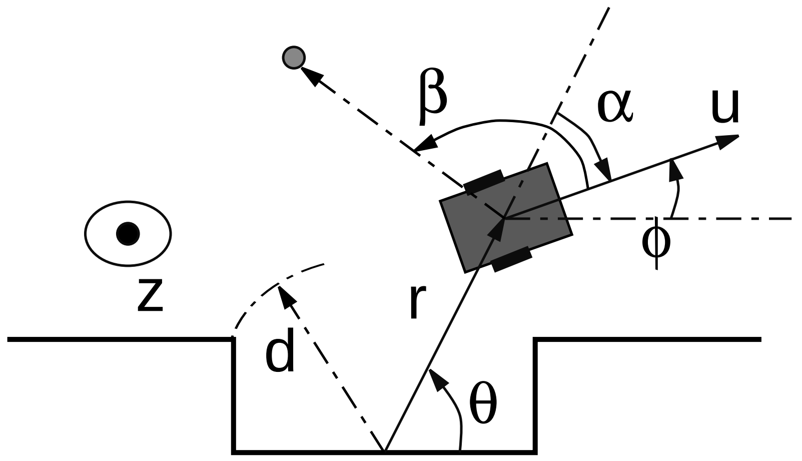
Thus if $\lim_{t\to\infty} u(t) \neq 0 \Longrightarrow \lim_{t\to\infty} r(t) = d$

Comments

$$\frac{\cos\alpha}{\left(\frac{\pi}{2}-\alpha\right)}$$



Linear velocity control



$$\omega = \frac{u}{r}\sin\alpha + \gamma\left(\frac{\pi}{2} - \alpha\right) + $$
$$- \; h\,u\,(d - r)\,\frac{\cos\alpha}{\frac{\pi}{2} - \alpha} \;:\; \gamma, h > 0$$

$$u = u_{max}\cos\beta \;:\; u_{max} > 0$$

Notice that all the states used for feedback are directly measured. There is no need for global self-localization.

At equilibrium $\omega = \frac{u}{d}$, i.e. the robot drives on the circle.

Comments

The law guarantees asymptotical, almost-global stability in the domain $\mathcal{D} = \left\{(\alpha, \theta, r)^T \in R^{3\times 1} : r \neq 0\right\}$.

The positive gains γ and h are open design parameters.

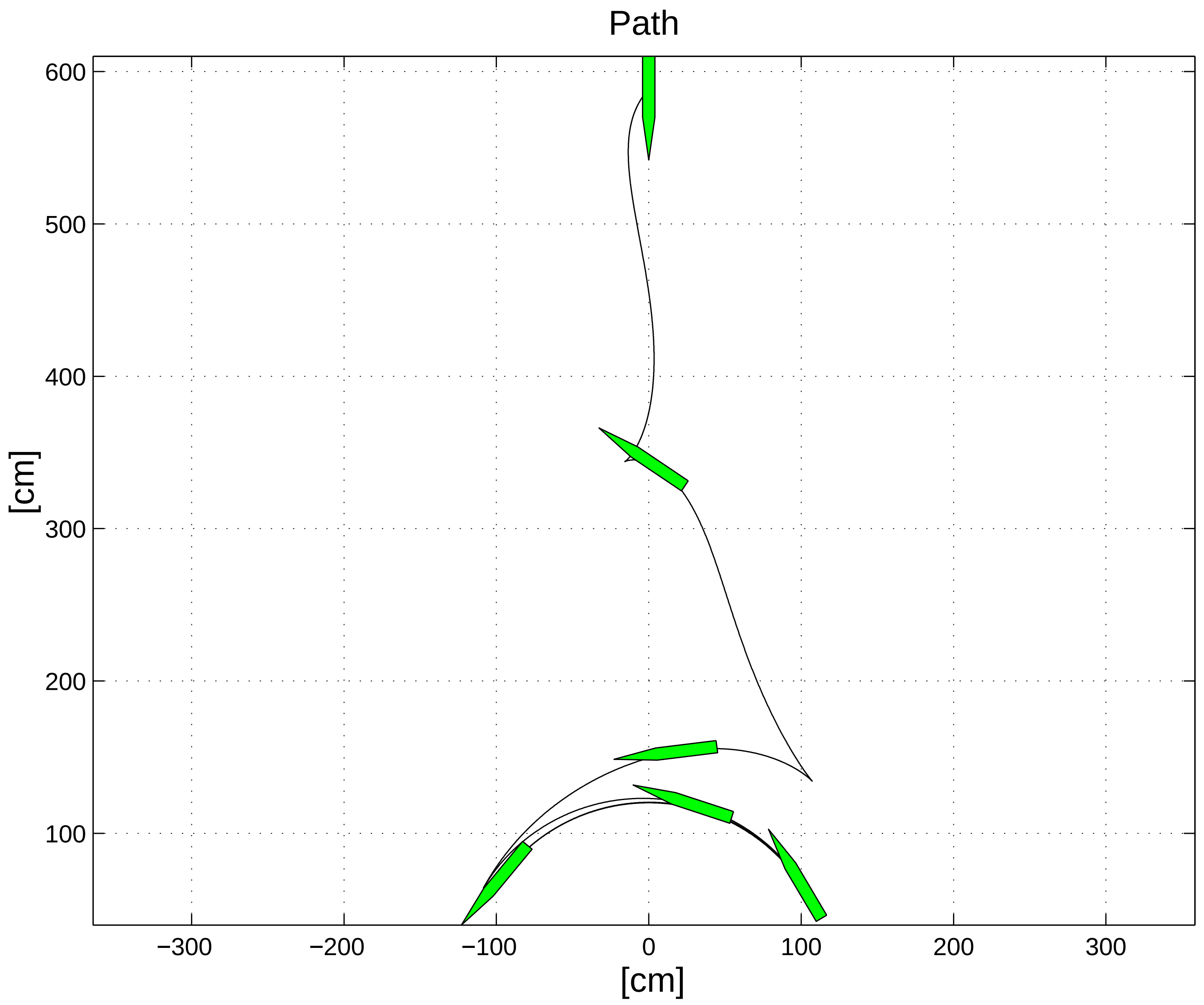
The linear velocity must be nonnull, but can be either positive or negative.

This same guidance law can be used to "home" the goal keeper when distant from its own goal. Of course the behaviour will depend upon the choice for u.

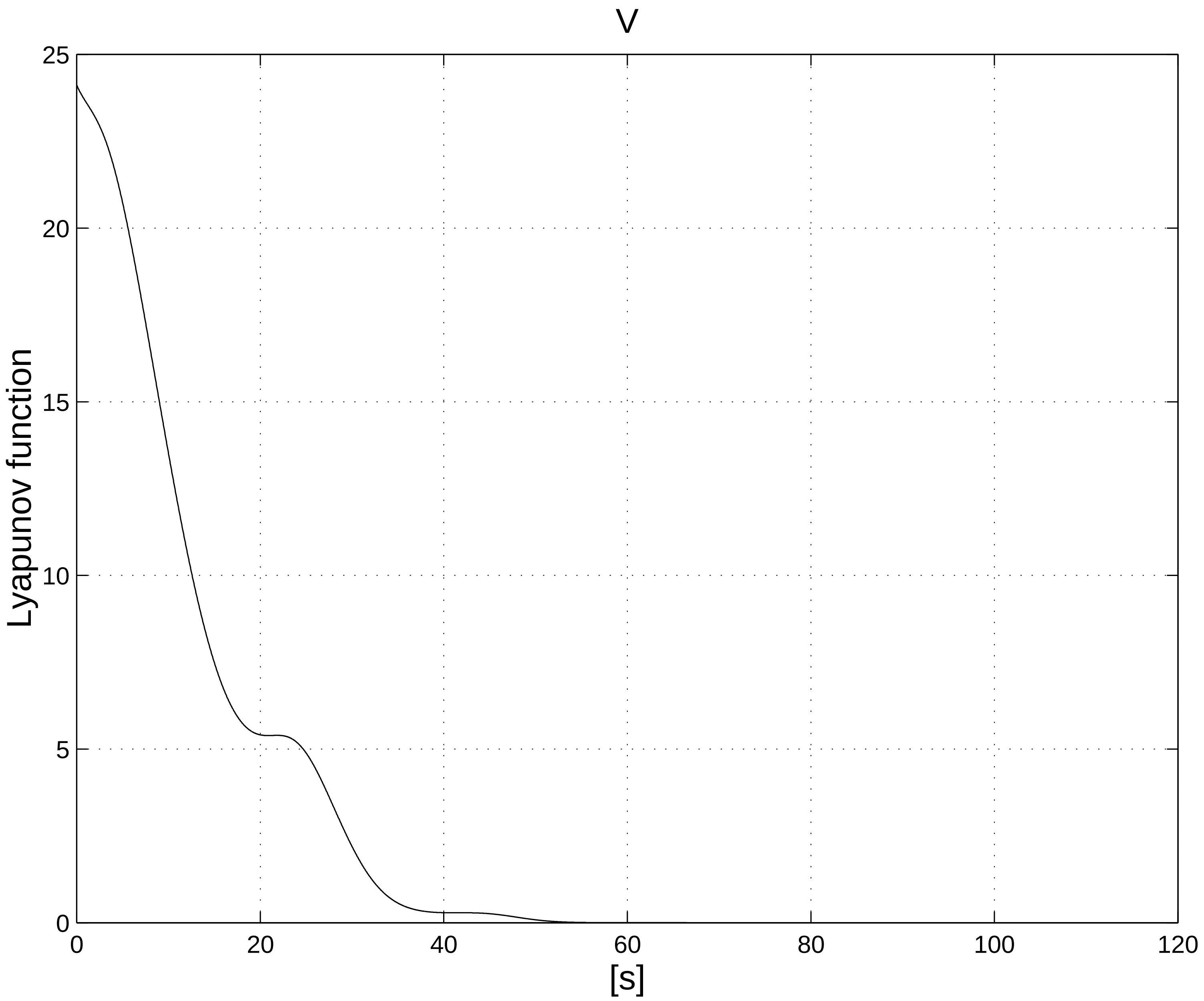
Simulations

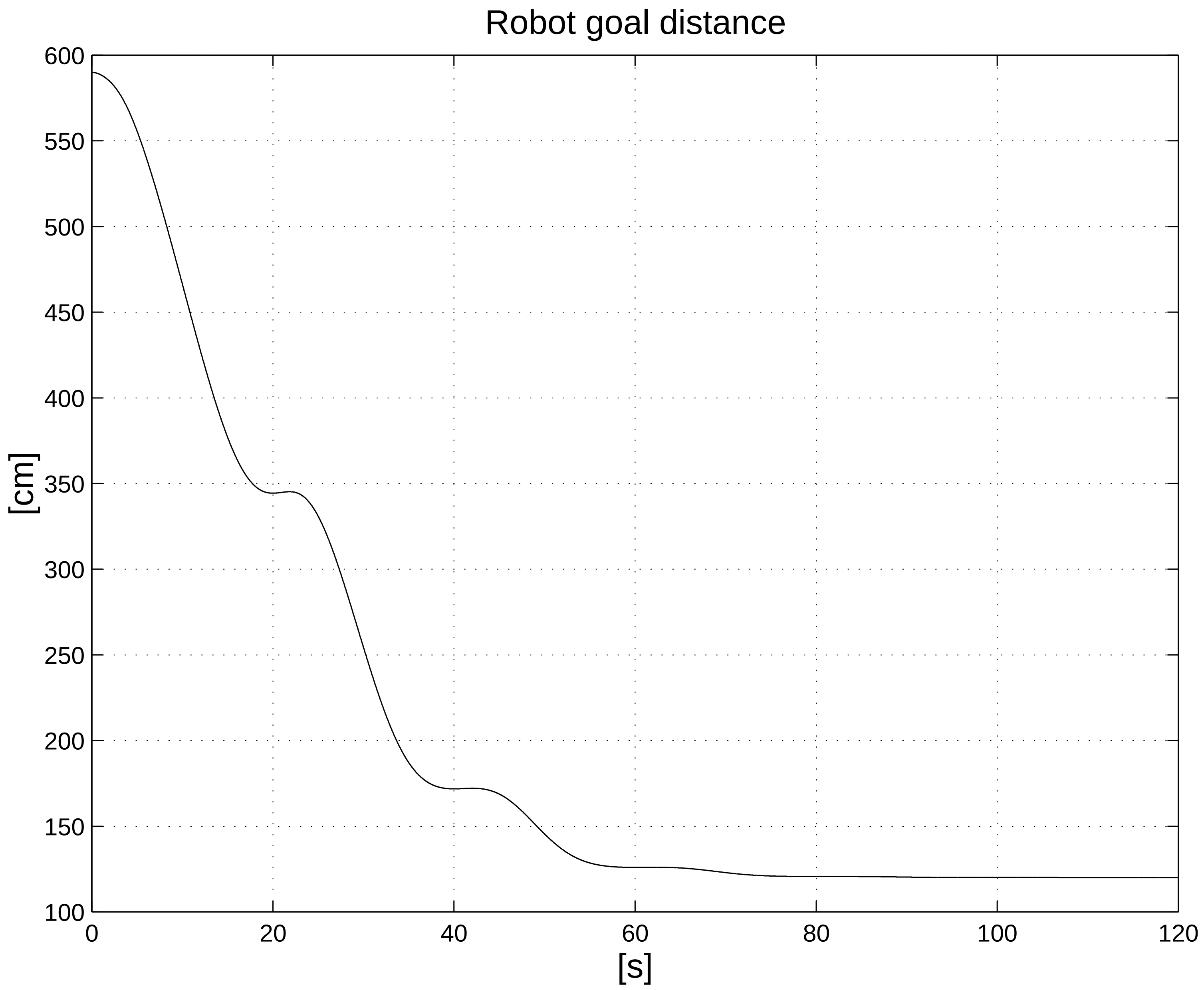
The first simulation refers to:

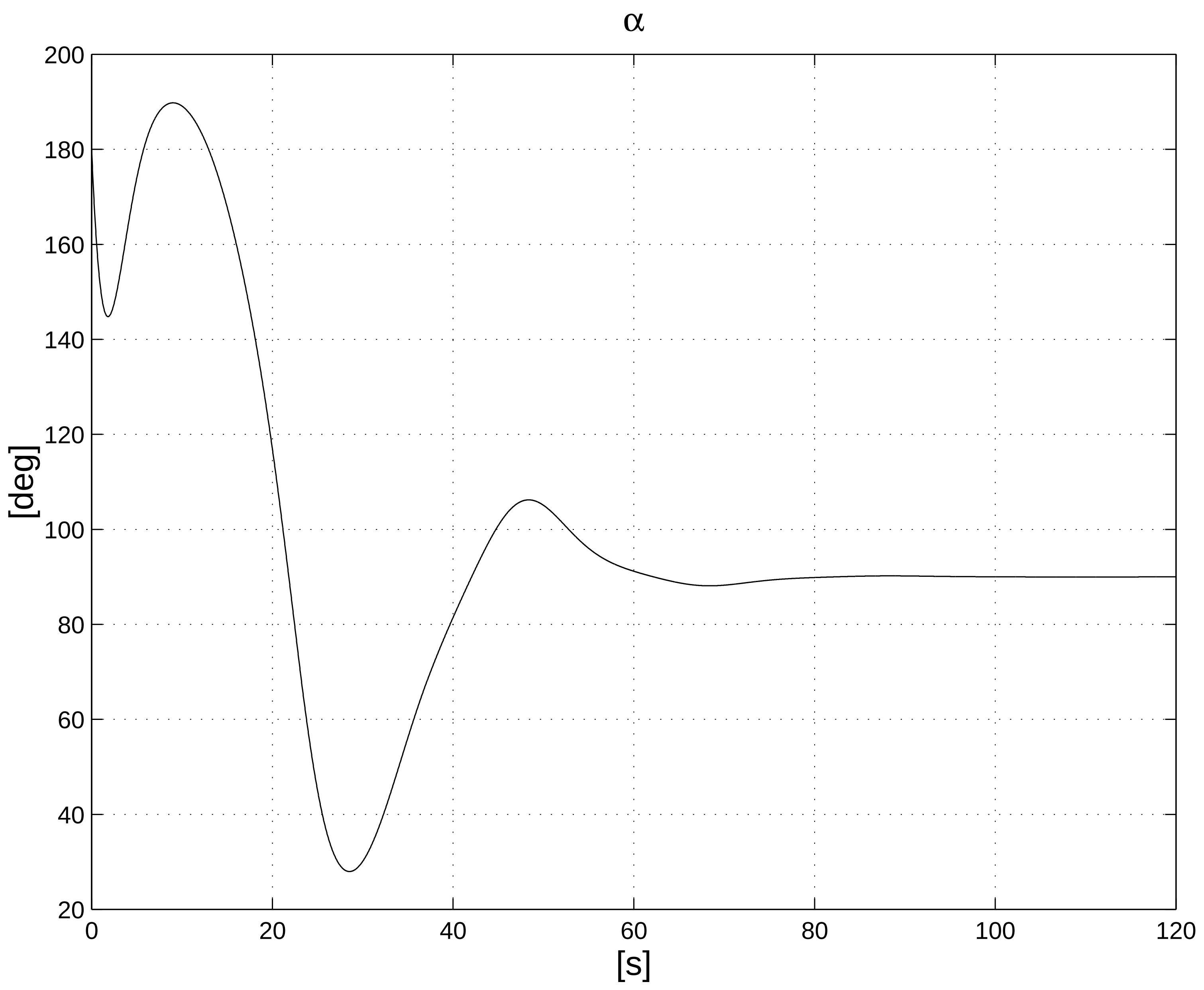
$$\begin{aligned} u &= u_{max} \sin ft \; [cm/s] \\ \gamma &= 1/2 \\ h &= 2.17 \cdot 10^{-4} \\ \alpha_0 &= \pi \\ \theta_0 &= \pi/2 \end{aligned}$$

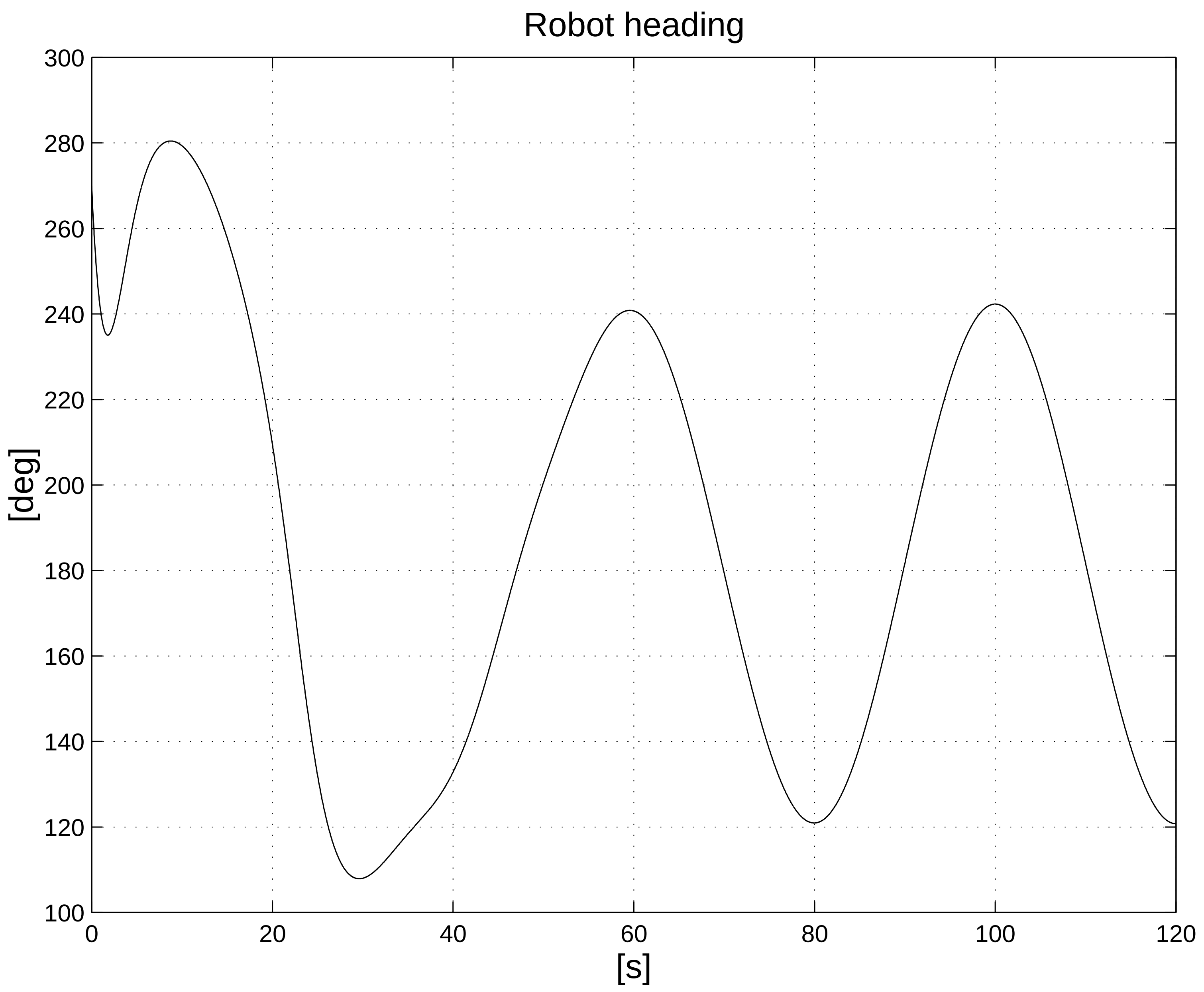


Path
600
500
400
300
200
100
[cm]
−300
−200
−100
0
100
200
300
[cm]

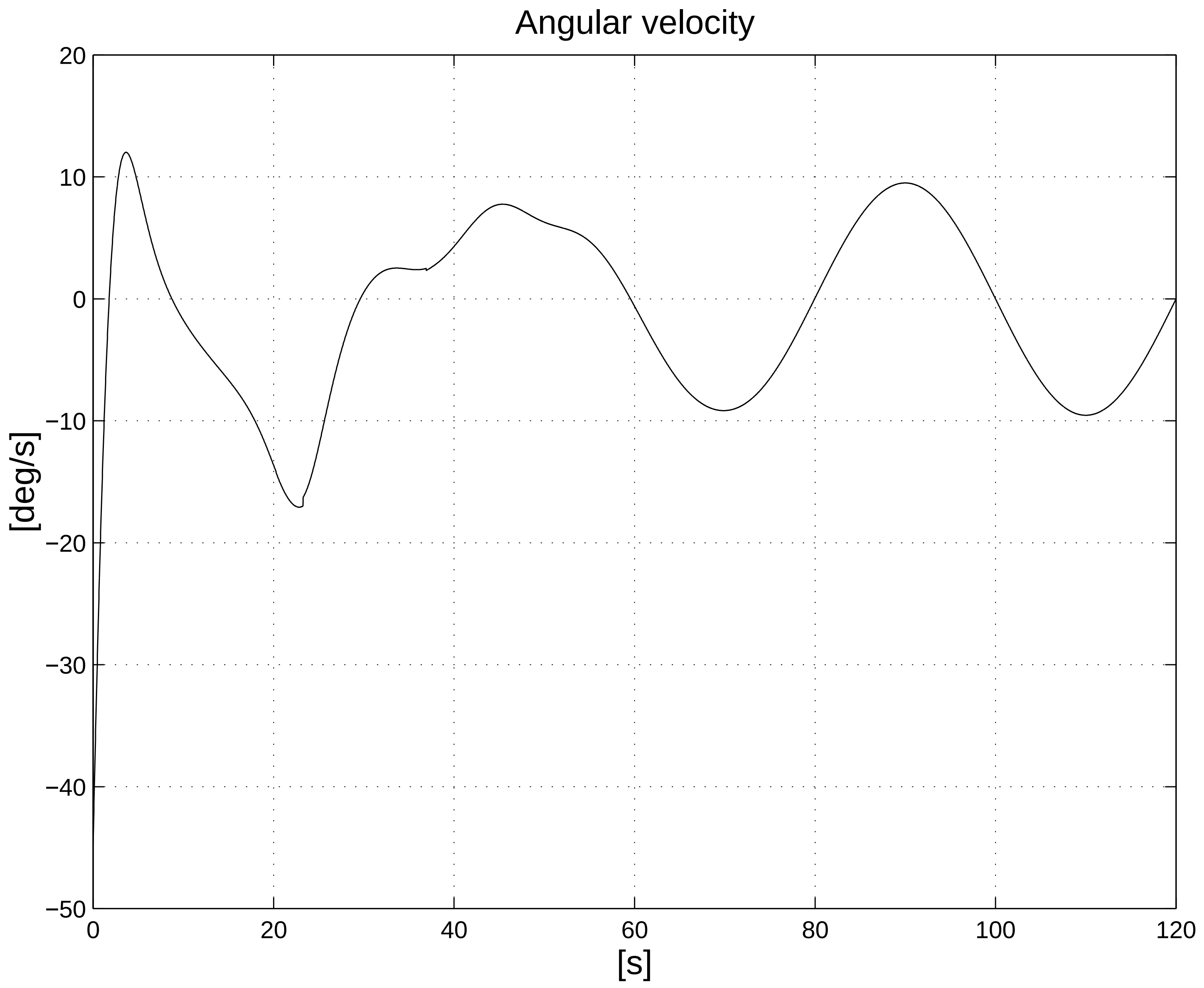

V
Lyapunov function
[s]
0
5
10
15
20
25
0
20
40
60
80
100
120


Robot goal distance
600
550
500
450
400
350
300
250
200
150
100
[cm]
0
20
40
60
80
100
120
[s]

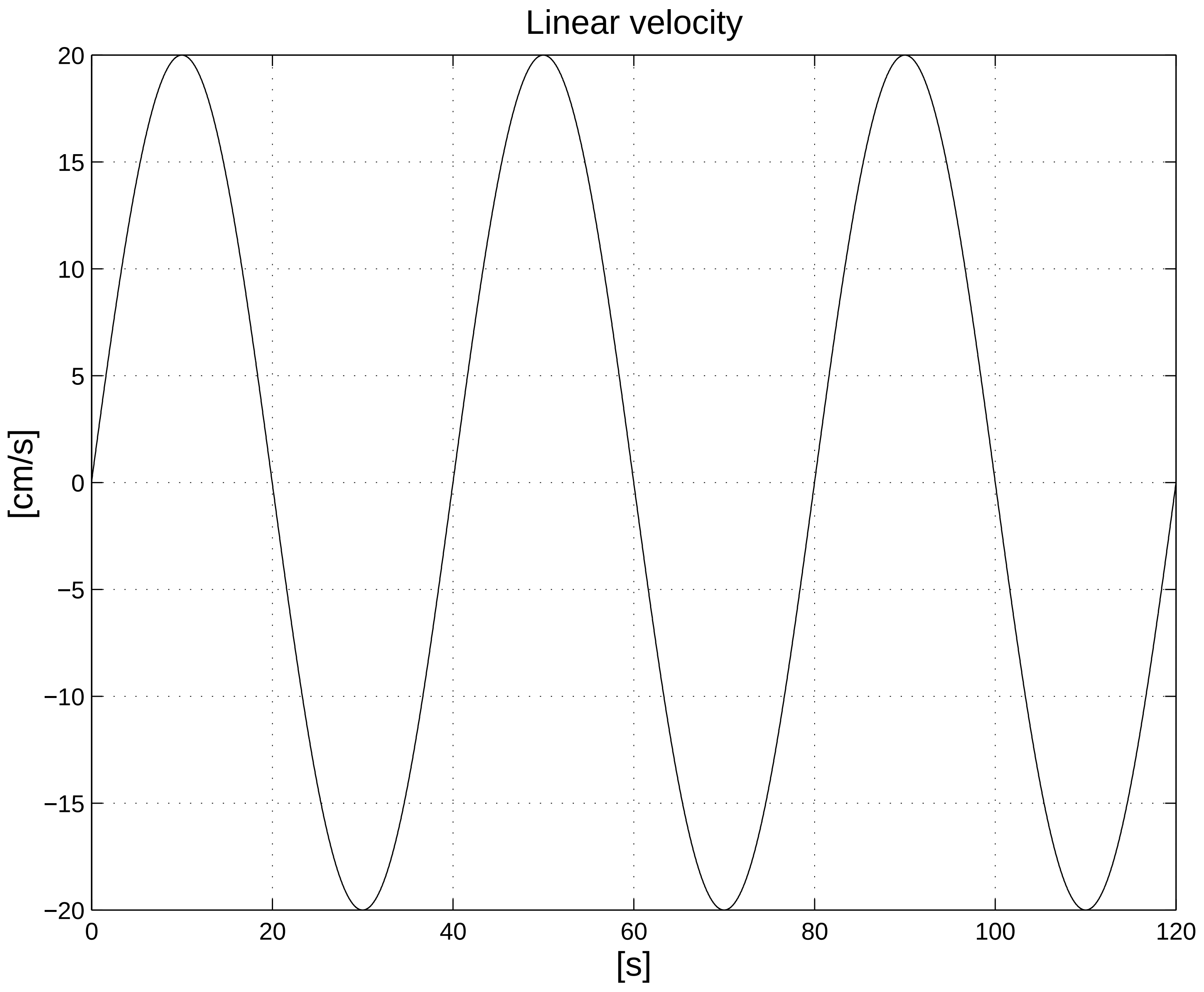

α
[deg]
[s]
200
180
160
140
120
100
80
60
40
20
0
20
40
60
80
100
120



Robot heading
300
280
260
240
220
200
180
160
140
120
100
[deg]
0
20
40
60
80
100
120
[s]



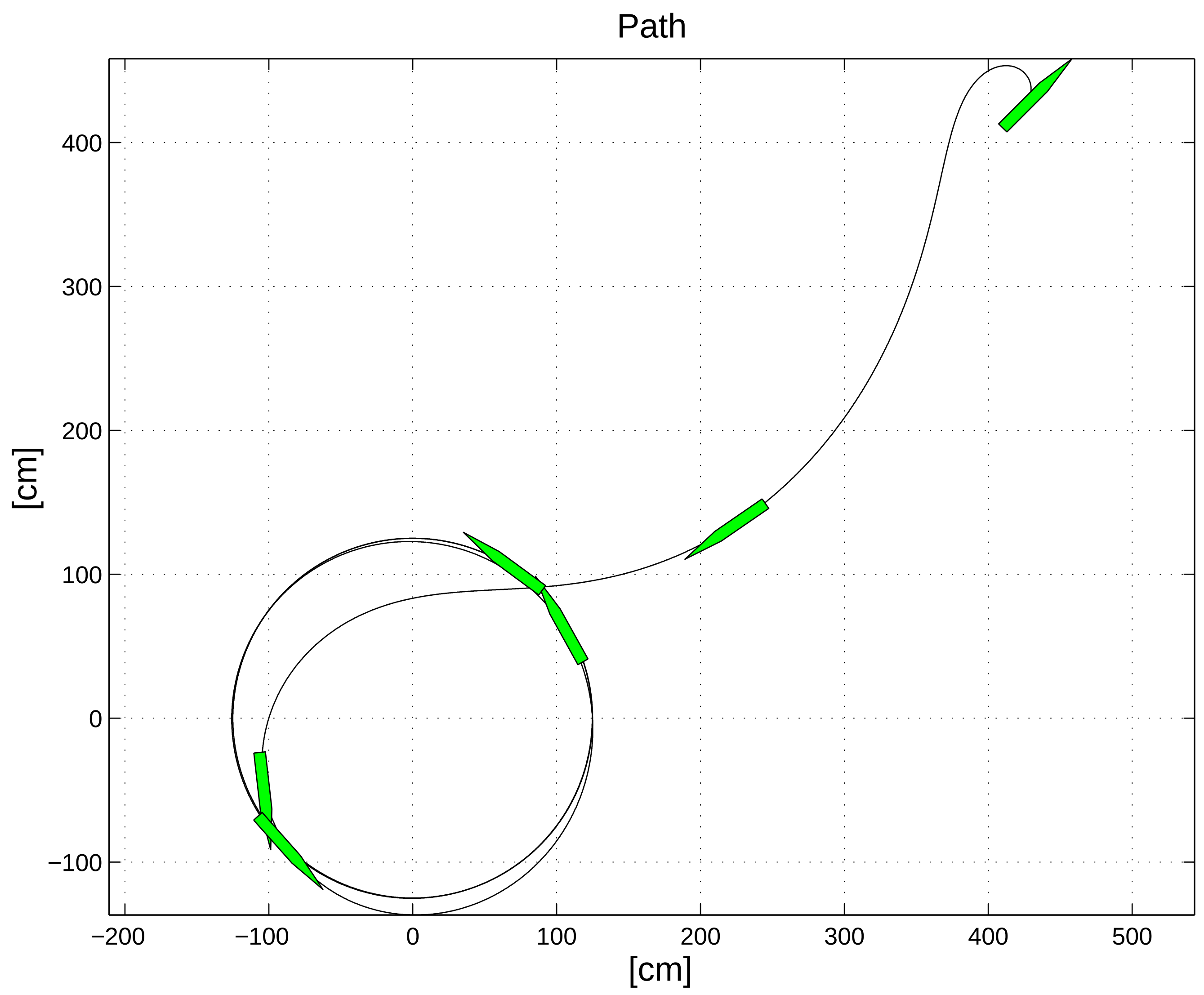
Angular velocity
[deg/s]
[s]
20
10
0
−10
−20
−30
−40
−50
0
20
40
60
80
100
120

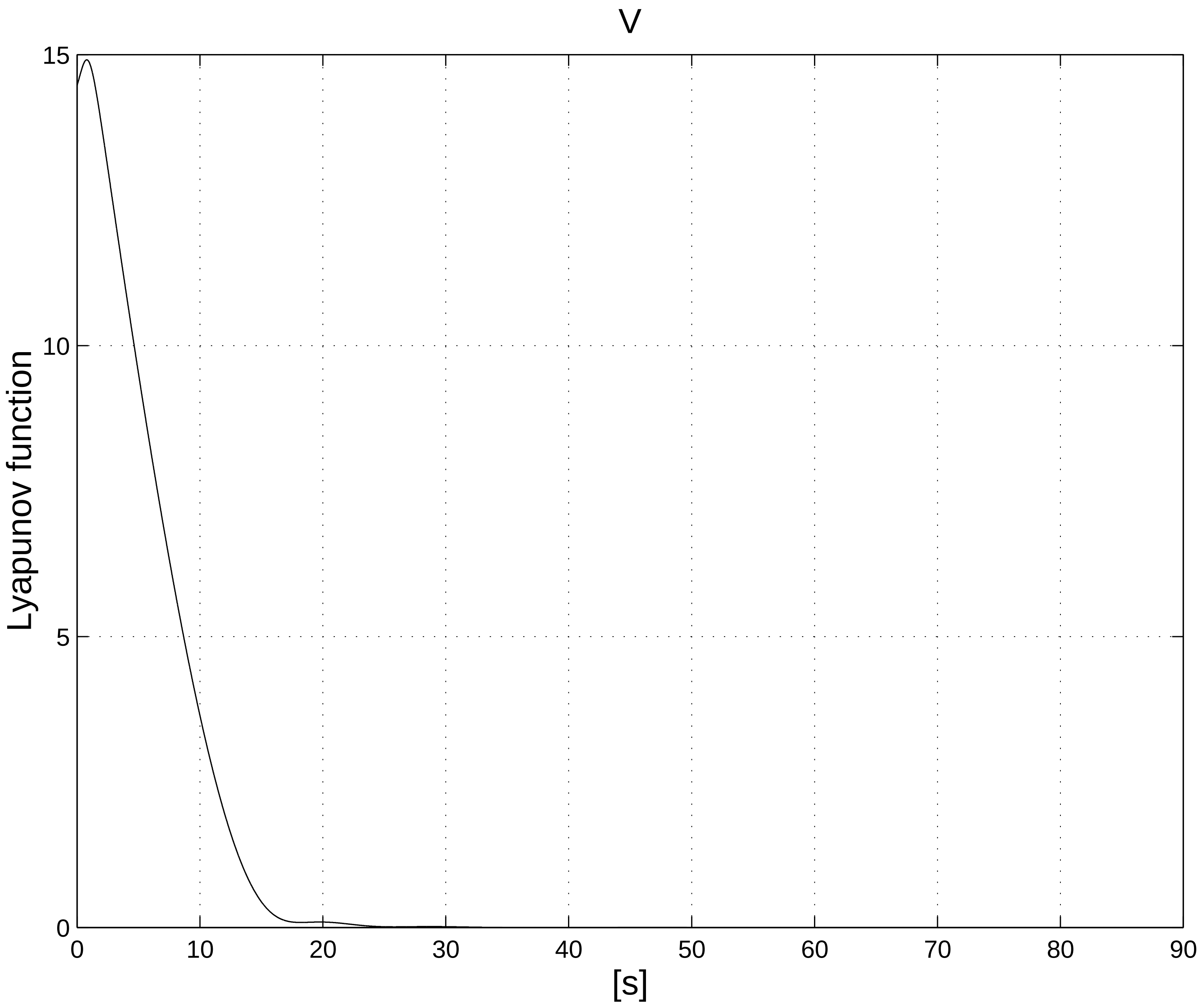

Linear velocity
20
15
10
5
0
−5
−10
−15
−20
[cm/s]
0
20
40
60
80
100
120
[s]

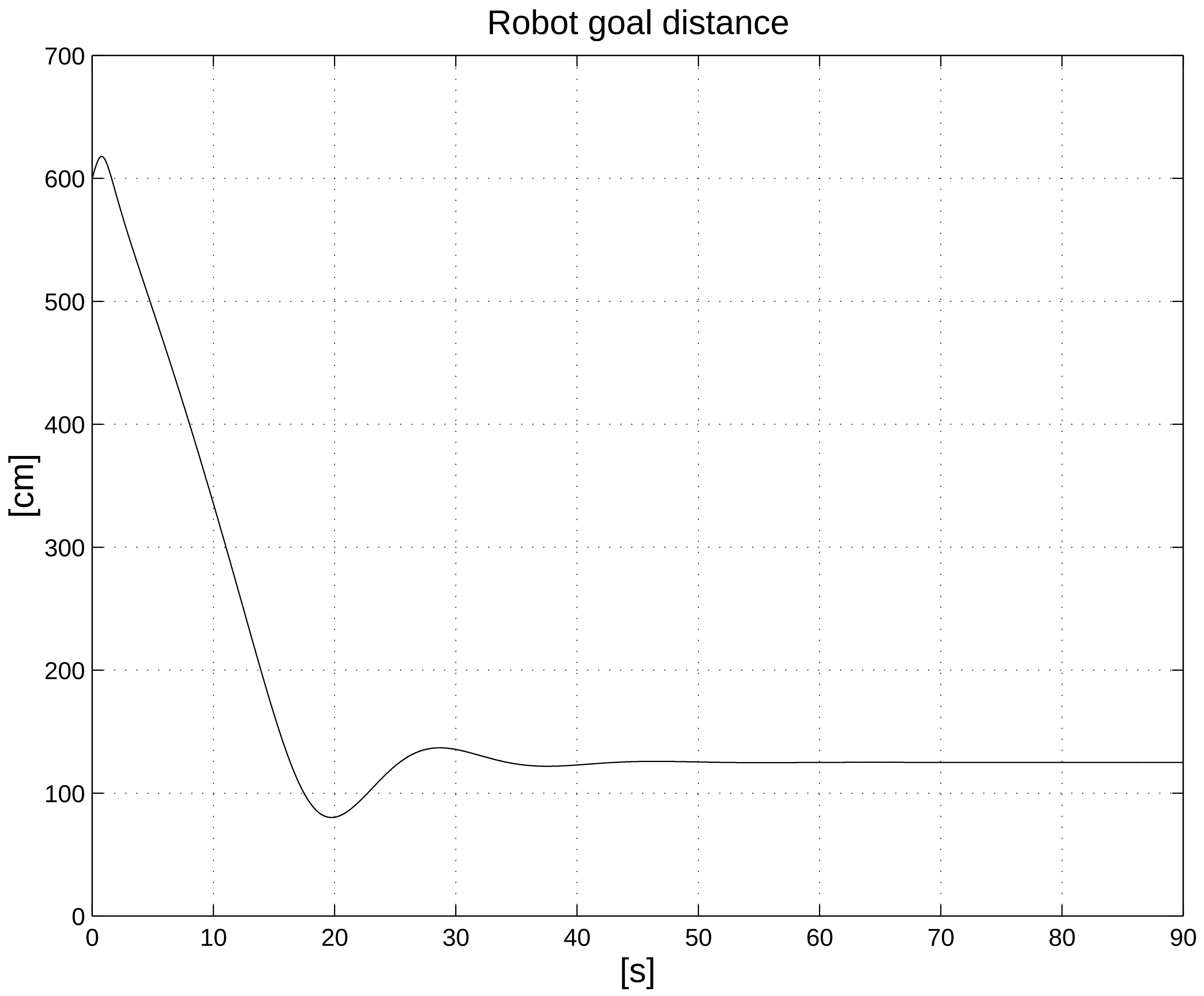
Simulations

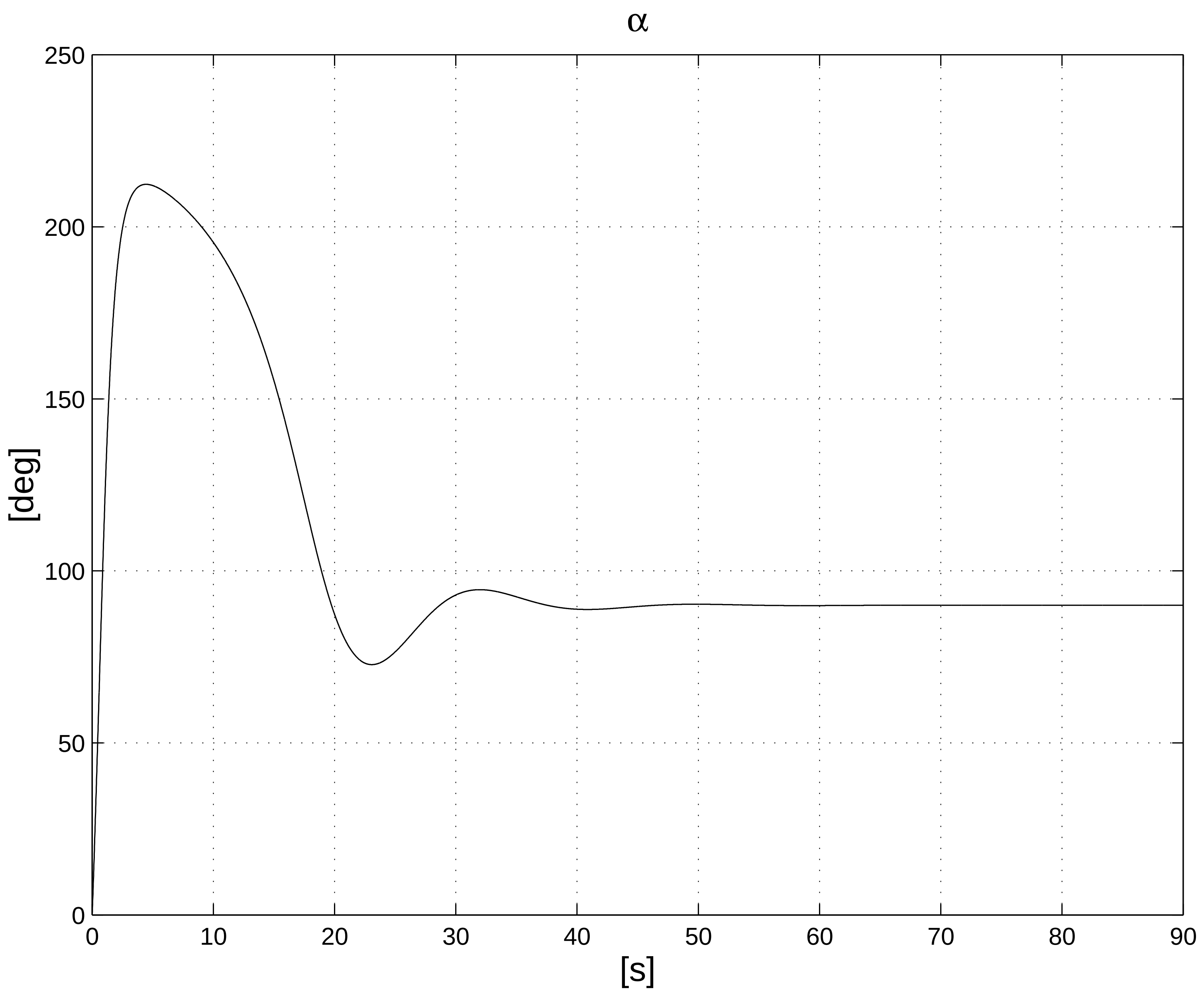
The second simulation refers to:

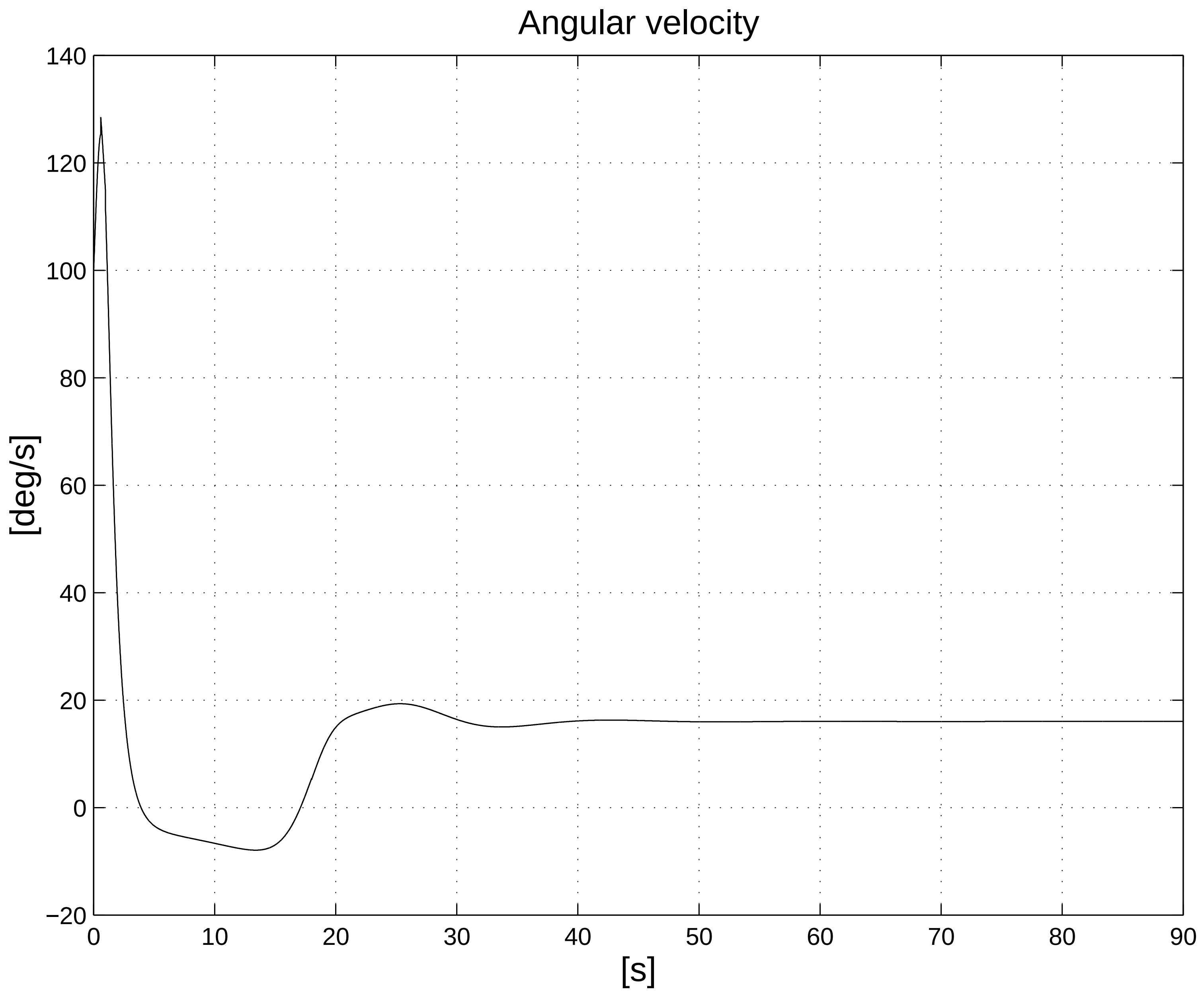
$$\begin{aligned} u &= 35[cm/s] \\ \gamma &= 1/2 \\ h &= 1.2 \cdot 10^{-4} \\ \alpha_0 &= 0 \\ \theta_0 &= \pi/4 \end{aligned}$$

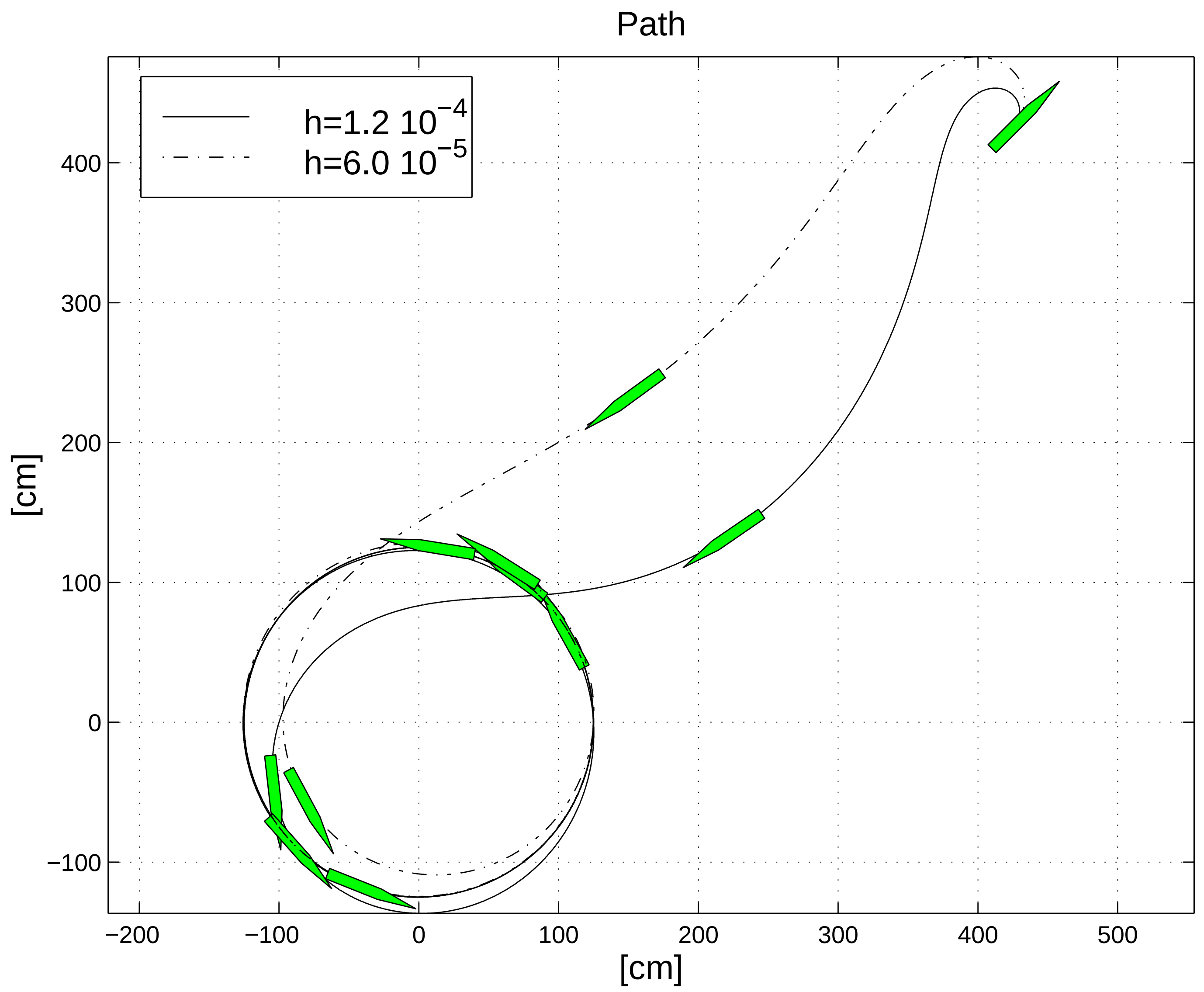

Path
[cm]
[cm]
400
300
200
100
0
−100
−200
−100
0
100
200
300
400
500

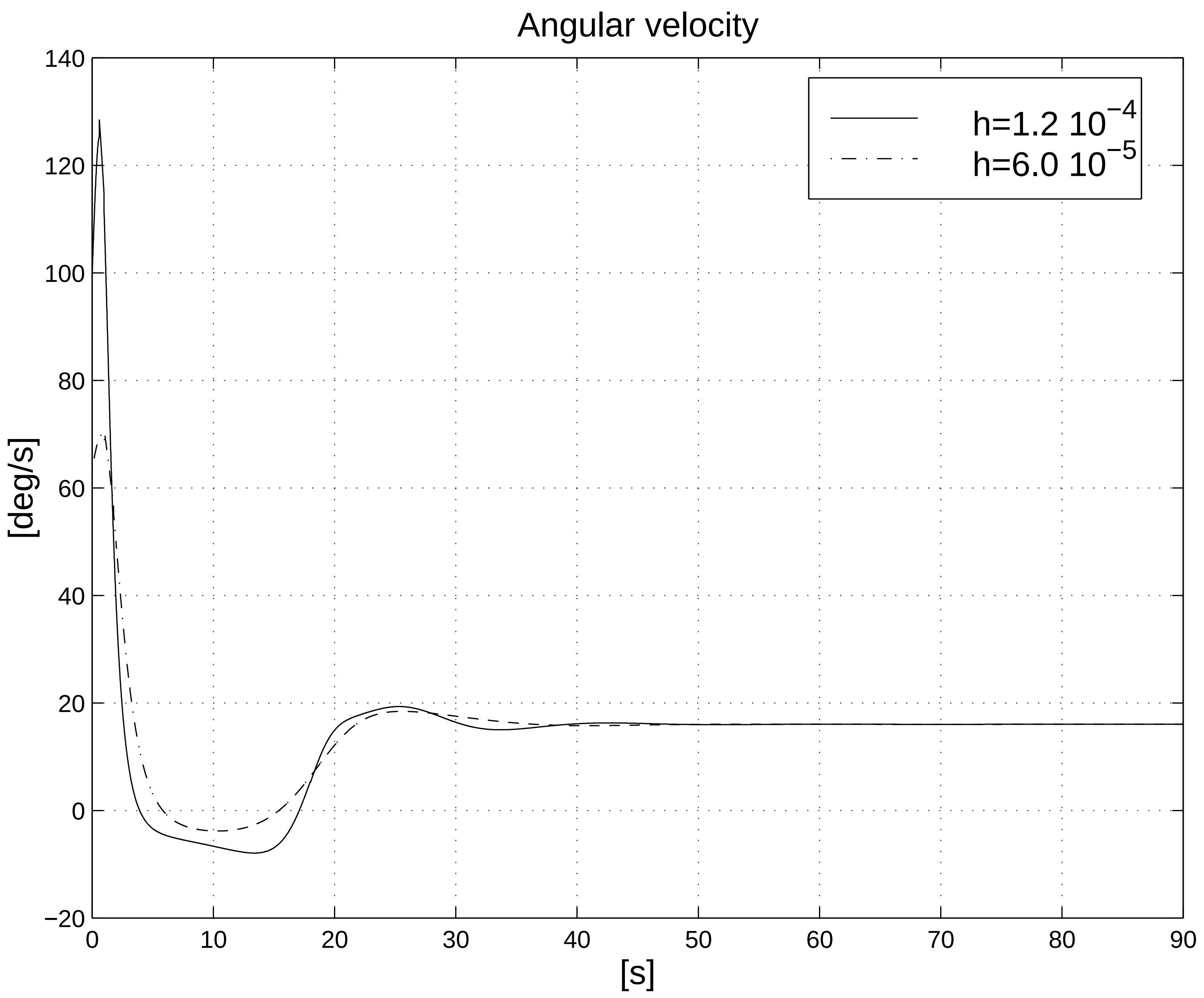

V
Lyapunov function
[s]
0
5
10
15
0
10
20
30
40
50
60
70
80
90


Robot goal distance
[cm]
[s]
0
100
200
300
400
500
600
700
0
10
20
30
40
50
60
70
80
90


α
[deg]
[s]
0
50
100
150
200
250
0
10
20
30
40
50
60
70
80
90


Angular velocity
[deg/s]
[s]
140
120
100
80
60
40
20
0
−20
0
10
20
30
40
50
60
70
80
90


Path
h=1.2 10^−4
h=6.0 10^−5
[cm]
[cm]
400
300
200
100
0
−100
−200
−100
0
100
200
300
400
500


Angular velocity
h=1.2 10^−4
h=6.0 10^−5
[deg/s]
[s]
140
120
100
80
60
40
20
0
−20
0
10
20
30
40
50
60
70
80
90

Concluding remarks

Additional design degrees of freedom can be added by noting that h does not need to be constant, but can be chosen to be any C^2 function of time such that

$$h > 0 \quad \cup \quad \dot{h} < 0$$

This approach allows to avoid planning *and* global self-localization at the expense of having to solve the obstacle avoidance issue with some heuristic ad hoc reactive behaviour that cannot guarantee the same performances obtained, at least in principle, otherwise.

Acknowledgments

A warm thank you to Thorsten Buchheim, Paul Levi, Daniele Nardi, The FhG-AiS BE-Team, Alessandro De Luca, Eliana, Martina

and to all of you for your attention.