How I Combined State Estimation, Passivity and Trajectory Optimization thanks to the

ADLipedia

Dr. Paolo Robuffo Giordano

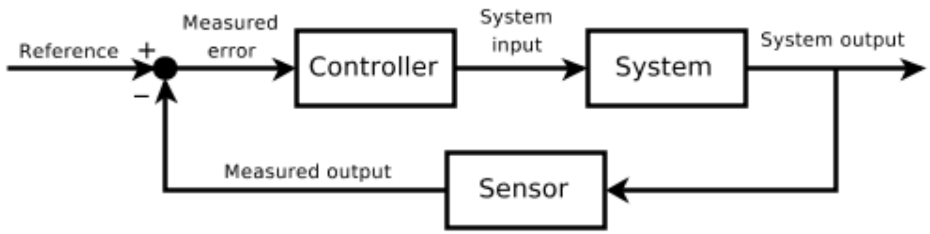
ADL Festschrift

January 9th, 2018

DIAG, Sapienza University of Rome

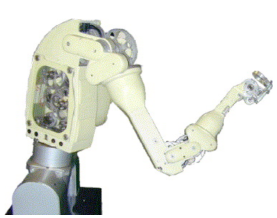
Some Personal History with ADL

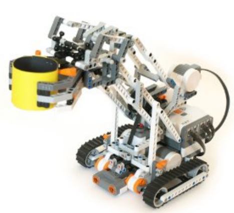
- 1999: Controlli Automatici (Automatic Control)



(**full first week** about motion control of a washing machine motor…)

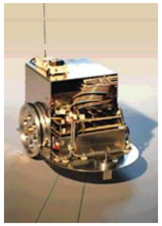
- 2000: Robotica Industriale (Robotics)

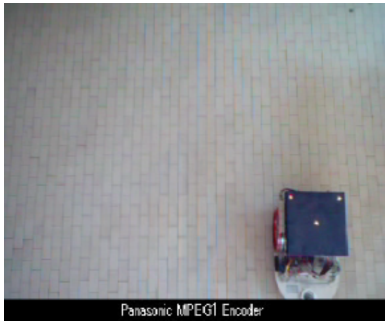


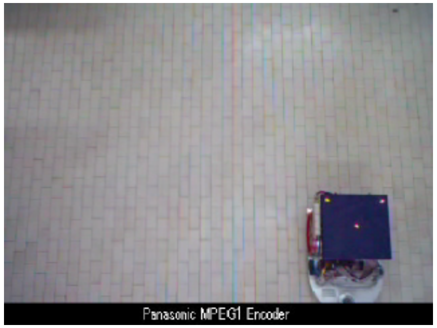


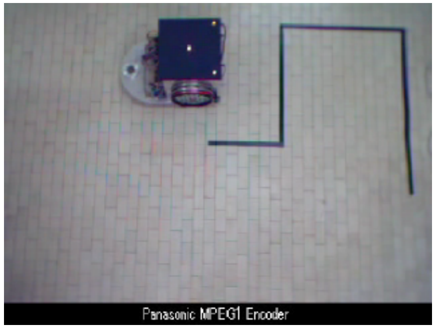
Some Personal History with ADL

- 2001: M.Sc. Thesis

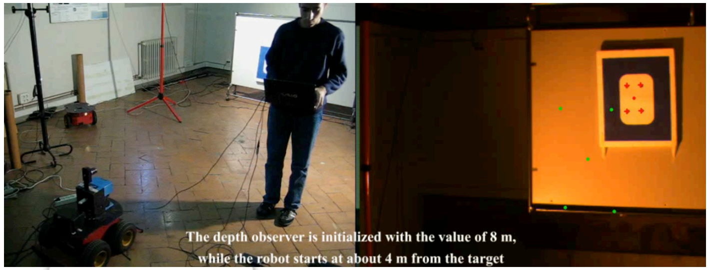




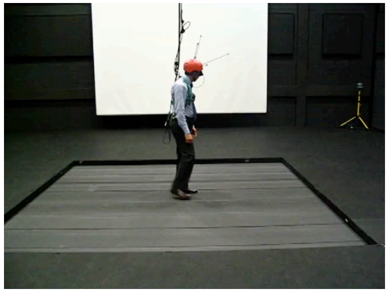




- 2004 – 2007 PhD



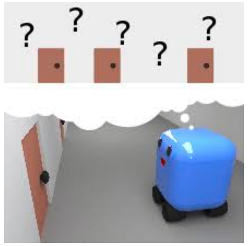
DLR



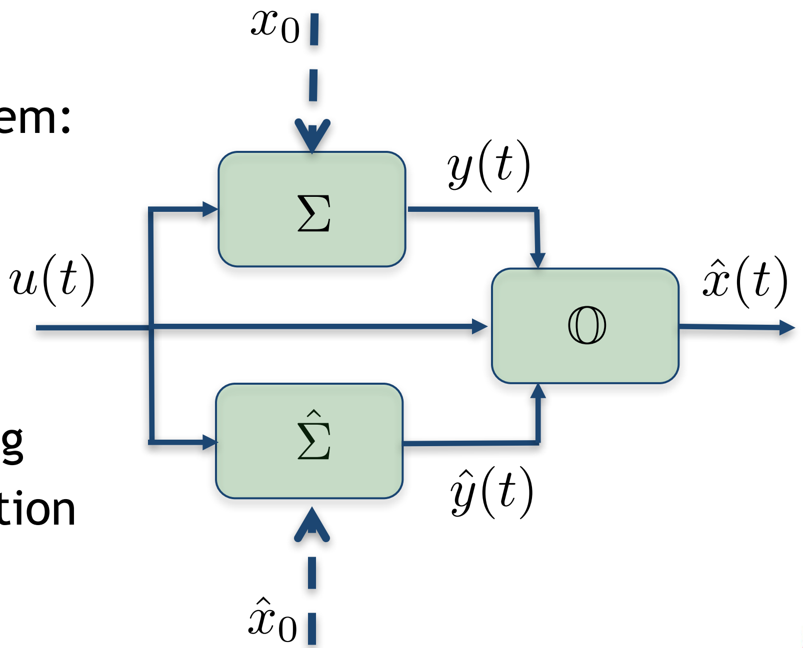
State Estimation

- **State estimation** is a classical problem in control theory and applications

- and Robotics is a good source of applications
  - Partial (and noisy) knowledge of the environment from onboard sensors
  - Need to recover the ‘world state’ in order to plan, act, reason, …

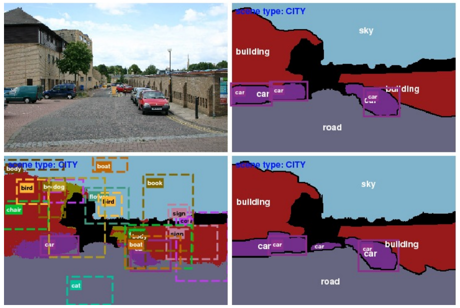


- Typical closed-loop scheme for a dynamical system:
  - known inputs $u(t)$
  - model $\hat{\Sigma}$ of the real plant Σ
  - known (measured) output $y(t)$
  - some update rule $\mathbb{O}$ which combines everything all together for producing a converging estimation $e(t) = x(t) - \hat{x}(t) \to 0$



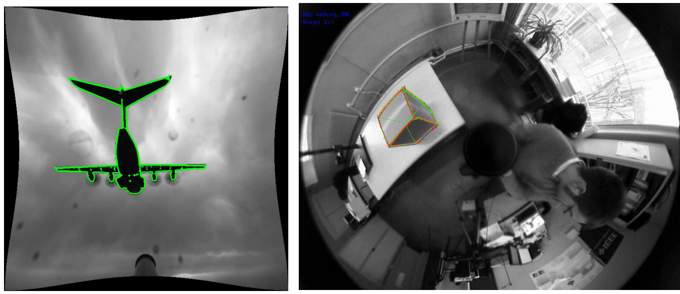
(Active) Structure from Motion

- Vision (cameras): extremely powerful but also complex sensing modality

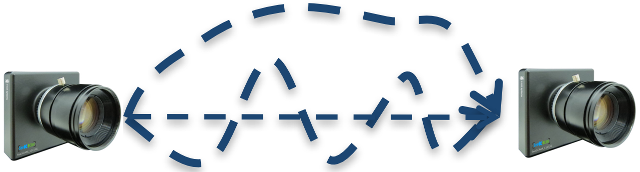


[Yao, et al., CVPR 2012]

- Many challenges to exploit vision in real-world robotics contexts
  - Scene understanding/classification
  - Visual tracking
  - Robust image processing (e.g., light conditions)
  - …
  - sensor mapping (perspective projection) $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix}$
    - nonlinear and non-injective


[Petit et al, ICRA 2014]
[Caron et al, RAS 2010]

- Structure from Motion (**SfM**): recover the missing 3D structure from the observed images
  - depth of points, distance of planes, size of objects, scale of multi-robot bearing formations
  - **Nonlinear estimation problem**: performance/accuracy depend on the camera trajectory



Structure from Motion

- In **2006** I start working on SfM from a "control" perspective (here a "**modern**" 2017 formulation)

- SfM problems can be shown to obey the following general form

$$\begin{cases} \dot{\boldsymbol{s}} = \boldsymbol{f}_m(\boldsymbol{s}, \boldsymbol{\omega}) + \boldsymbol{\Omega}^T(\boldsymbol{s}, \boldsymbol{v})\boldsymbol{\chi} \\ \dot{\boldsymbol{\chi}} = \boldsymbol{f}_u(\boldsymbol{s}, \boldsymbol{\chi}, \boldsymbol{u}) \end{cases}$$

$\boldsymbol{s}$ → measured visual features
$\boldsymbol{\chi}$ → unknown 3D structure
$(\boldsymbol{v}, \boldsymbol{\omega})$ → camera linear/angular vel

- A general nonlinear observer for these systems can be built as

$$\begin{cases} \dot{\hat{\boldsymbol{s}}} = \boldsymbol{f}_m(\boldsymbol{s}, \boldsymbol{\omega}) + \boldsymbol{\Omega}^T(\boldsymbol{s}, \boldsymbol{v})\hat{\boldsymbol{\chi}} + \boldsymbol{H}\boldsymbol{\xi} \\ \dot{\hat{\boldsymbol{\chi}}} = \boldsymbol{f}_u(\boldsymbol{s}, \hat{\boldsymbol{\chi}}, \boldsymbol{u}) + \alpha\boldsymbol{\Omega}(\boldsymbol{s}, \boldsymbol{v})\boldsymbol{\xi} \end{cases}$$

$\boldsymbol{\xi} = \boldsymbol{s} - \hat{\boldsymbol{s}}$ → "prediction" error
$\boldsymbol{H}, \alpha$ → estimation gains

- This yields the estimation error dynamics

$$\begin{cases} \dot{\boldsymbol{\xi}} = -\boldsymbol{H}\boldsymbol{\xi} + \boldsymbol{\Omega}^T(\boldsymbol{s}, \boldsymbol{v})\boldsymbol{z} \\ \dot{\boldsymbol{z}} = -\alpha\boldsymbol{\Omega}(\boldsymbol{s}, \boldsymbol{v})\boldsymbol{\xi} + \boldsymbol{g}(\boldsymbol{z}, t) \end{cases}$$

$\boldsymbol{z} = \boldsymbol{\chi} - \hat{\boldsymbol{\chi}}$ → estimation error

$+\boldsymbol{g}(\boldsymbol{z}, t)$ → vanishing disturbance

$$\boldsymbol{g}(\boldsymbol{z}, t) = \boldsymbol{f}_u(\boldsymbol{s}, \boldsymbol{\chi}, \boldsymbol{u}) - \boldsymbol{f}_u(\boldsymbol{s}, \hat{\boldsymbol{\chi}}, \boldsymbol{u})$$

Structure from Motion

- Error dynamics

$$\begin{cases} \dot{\boldsymbol{\xi}} = -\boldsymbol{H}\boldsymbol{\xi} + \boldsymbol{\Omega}^T(\boldsymbol{s},\, \boldsymbol{v})\boldsymbol{z} \\ \dot{\boldsymbol{z}} = -\alpha\boldsymbol{\Omega}(\boldsymbol{s},\, \boldsymbol{v})\boldsymbol{\xi} + \boldsymbol{g}(\boldsymbol{z},\, t) \end{cases} \quad (\blacksquare)$$

- If $\boldsymbol{g} = \boldsymbol{0}$ then (■) GAS if the PE condition holds $\displaystyle\int_t^{t+T} \boldsymbol{\Omega}(\tau)\,\boldsymbol{\Omega}^T(\tau)\,d\tau \geq \gamma\boldsymbol{I} > 0$

- However, in most cases of interest $\boldsymbol{g} \neq \boldsymbol{0}$ (e.g., point features, image moments, plane parameters …)
  - Would need of an explicit Lyapunov function for (■) for characterizing stability when $\boldsymbol{g} \neq \boldsymbol{0}$

- ICRA 2007 deadline (September 15, 2016) is approaching fast
  - I am not able to find any Lyapunov function
  - ADL is busy (euphemism) with the preparations for ICRA 2007

- On September 14, 2016 (-1 day to ICRA 2007 deadline) I finally get an opening
  - In a 10-min break he looks at (■) and says something like **"it should probably be something related to passivity considerations…"**, and then goes back to ICRA paperwork
  - ..and I go back to my desk without any real clue of what to do

Structure from Motion

- **2013**: 7 years later I get back to this issue together with a Ph.D. student (R. Spica)

- It turns out that with a simple change of coordinates $\tilde{\boldsymbol{z}} = \boldsymbol{z}/\sqrt{\alpha}$ the error dynamics can be put in a “port-Hamiltonian form”

$$\begin{bmatrix} \dot{\boldsymbol{\xi}} \\ \dot{\tilde{\boldsymbol{z}}} \end{bmatrix} = \left(\begin{bmatrix} \mathbf{0} & \sqrt{\alpha}\boldsymbol{\Omega}^T \\ -\sqrt{\alpha}\boldsymbol{\Omega} & \mathbf{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{H} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \right) \begin{bmatrix} \boldsymbol{\xi} \\ \tilde{\boldsymbol{z}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \tilde{\boldsymbol{g}} \end{bmatrix}$$

TOLD YA

and the Lyapunov function is just the total energy $\mathcal{H}(\boldsymbol{\xi}, \boldsymbol{z}) = \frac{1}{2}\boldsymbol{\xi}^T\boldsymbol{\xi} + \frac{1}{2\alpha}\boldsymbol{z}^T\boldsymbol{z}$

- Matrix $\boldsymbol{\Omega}$ mediates the energy exchange between $\boldsymbol{\xi}$ (prediction error) and $\boldsymbol{z}$ (estimation error)

- Matrix $\boldsymbol{H}$ dissipates energy (on the “measurable” error $\boldsymbol{\xi}$)

- PE condition: keep the energy shuffling around (via $\boldsymbol{\Omega}$) so that $\boldsymbol{H}$ can dissipate…

Structure from Motion

- Further consequences: the error system can be seen as a mass-spring-damper system

$$\ddot{\boldsymbol{\eta}} = -\boldsymbol{D}_1\dot{\boldsymbol{\eta}} - \alpha \boldsymbol{S}^2\boldsymbol{\eta} \qquad \boldsymbol{H} = \boldsymbol{V}\begin{bmatrix} \boldsymbol{D}_1 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{D}_2 \end{bmatrix}\boldsymbol{V}^T \qquad \boldsymbol{\Omega} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$$

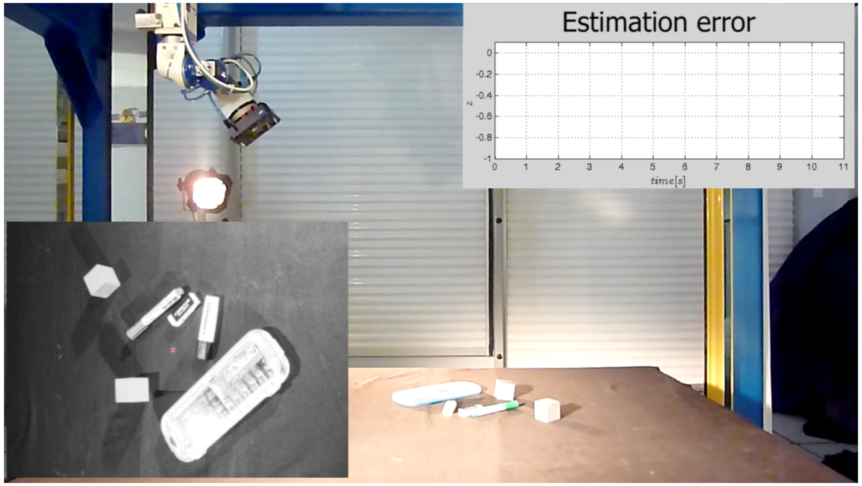
where $\boldsymbol{D}_1$ is a desired damping term and $\boldsymbol{S}^2 = \mathrm{diag}(\sigma_i^2)$ (eigenvalues of $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$) is the "stiffness matrix"

- $\boldsymbol{\Omega}(t) = \boldsymbol{\Omega}(\boldsymbol{s},\, \boldsymbol{v}) \longrightarrow \sigma_i^2(t) = \sigma^2(\boldsymbol{s},\, \boldsymbol{v})$: possibility to act on $\boldsymbol{v}$ to control/assign a desired dynamics to the estimation error (like tuning the damping/stiffness in interaction control)

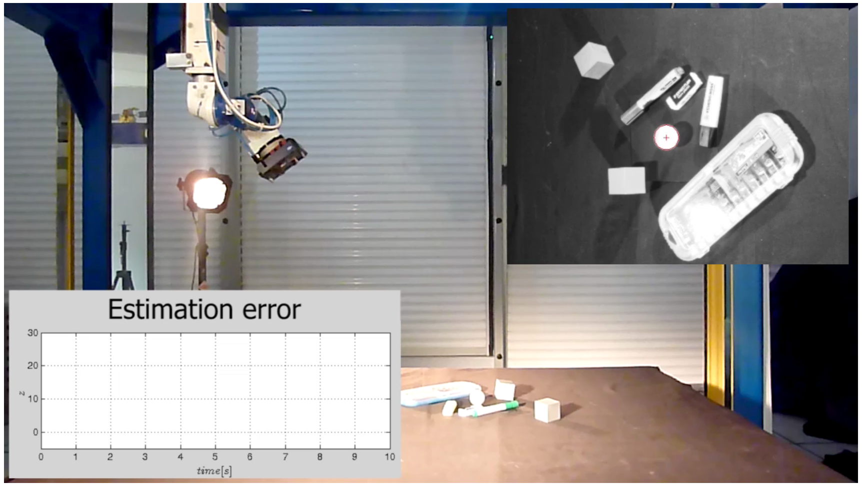
- For instance $\dot{\boldsymbol{v}} = \dfrac{k_1 \boldsymbol{v}}{\|\boldsymbol{v}\|^2}\left(\|\boldsymbol{v}_0\| - \|\boldsymbol{v}\|\right) + k_2\left(\boldsymbol{I}_3 - \dfrac{\boldsymbol{v}\boldsymbol{v}^T}{\|\boldsymbol{v}\|^2}\right)\nabla_{\boldsymbol{v}}\sigma_1^2$

- Optimize the camera motion direction while keeping a constant linear velocity norm

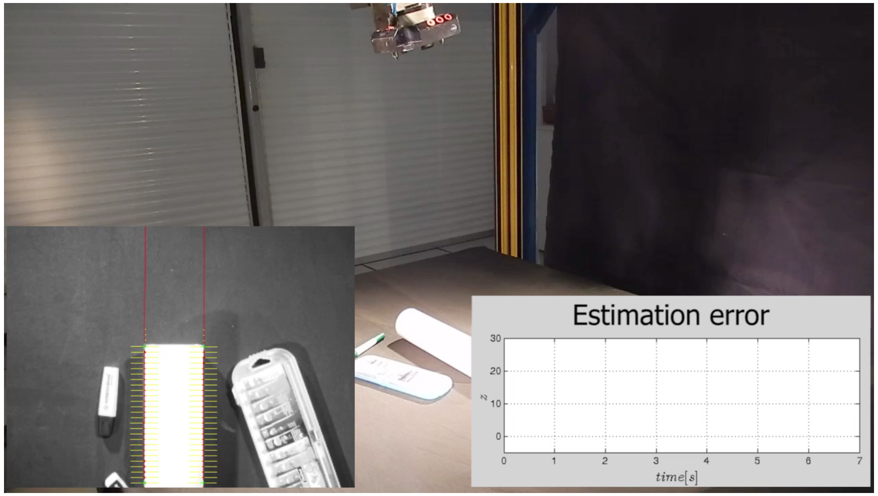
Active Visual State Estimation



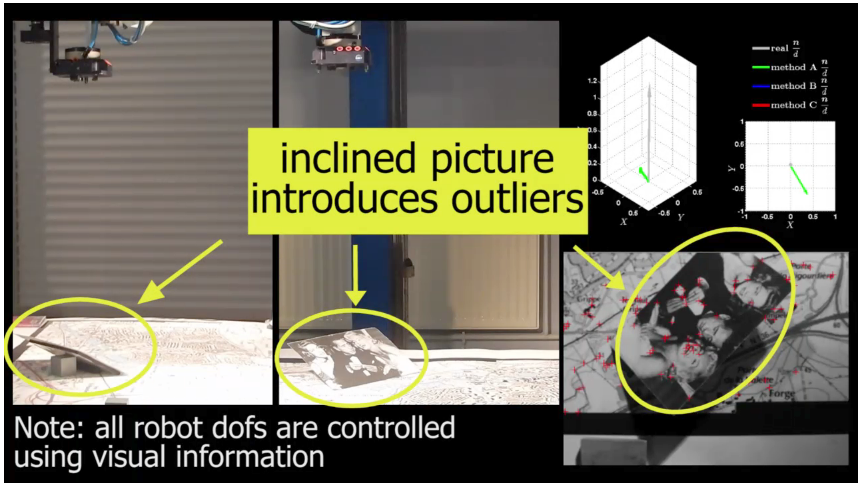
point [CDC'13, TRO'14]



sphere [ICRA'14, TRO'14]



cylinder [ICRA'14, TRO'14]



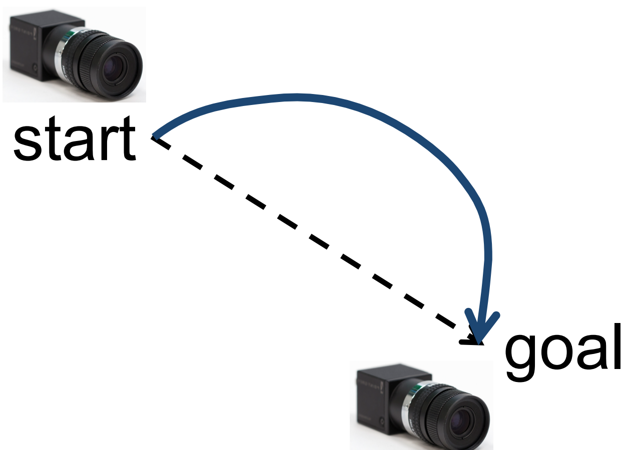
plane [CDC'13, ICRA'14, ICRA'15]

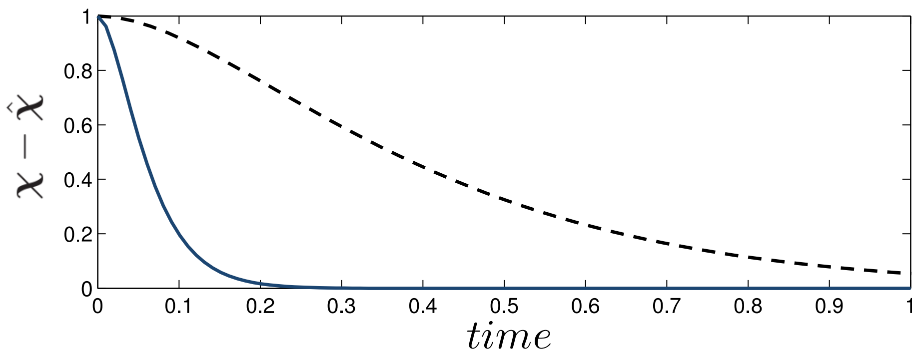
CDC 2013, ICRA 2014,
ICRA 2015, TRO 2014

Coupling with Visual Control

- Further developments: how to plug the active estimation in the execution of a visual task

- The camera should arrive at a desired location while following an "informative path" for concurrently reconstructing the 3D scene (which is used by the controller)





- Benefits:
  - Better knowledge of the scene during task execution
    → task convergence closer to ideality
  - Better knowledge of the scene at the end of the task
    → can be used for other purposes

Coupling with Visual Control

- Cost function representative of the estimation excitation $\mathcal{V} = \mathcal{V}(\sigma_i^2(\boldsymbol{s},\, \boldsymbol{v}))$ (to be maximized)

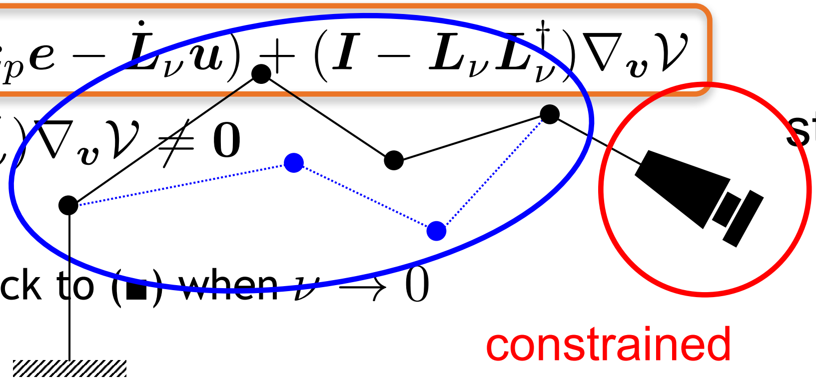
- Second-order redundancy resolution $\dot{\boldsymbol{s}} = \boldsymbol{L_s u}$ → $\ddot{\boldsymbol{s}} = \boldsymbol{L_s}\dot{\boldsymbol{u}} + \dot{\boldsymbol{L}}_{\boldsymbol{s}}\boldsymbol{u}$

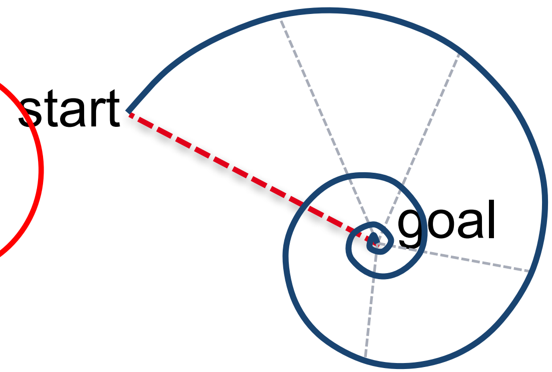
$$\dot{\boldsymbol{u}} = \boldsymbol{L}_{\boldsymbol{s}}^{\dagger}(-k_v\dot{\boldsymbol{e}} - k_p\boldsymbol{e} - \dot{\boldsymbol{L}}_{\boldsymbol{s}}\boldsymbol{u}) + (\boldsymbol{I} - \boldsymbol{L_s L_s^{\dagger}})\nabla_{\boldsymbol{v}}\mathcal{V} \quad (\blacksquare) \qquad \boldsymbol{e} = \boldsymbol{s} - \boldsymbol{s}_d$$

- However, in any reasonable case $(\boldsymbol{I} - \boldsymbol{L_s L_s^{\dagger}})\nabla_{\boldsymbol{v}}\mathcal{V} = \boldsymbol{0}$ (no room for optimization of camera motion)
- Better results by choosing to regulate the norm of the task error $\nu = \|\boldsymbol{e}\|$ $\qquad \boldsymbol{L}_\nu = \dfrac{\boldsymbol{e}^T\boldsymbol{L_s}}{\nu}$

$$\dot{\boldsymbol{u}} = \boldsymbol{L}_{\nu}^{\dagger}(-k_v\dot{\boldsymbol{e}} - k_p\boldsymbol{e} - \dot{\boldsymbol{L}}_{\nu}\boldsymbol{u}) + (\boldsymbol{I} - \boldsymbol{L}_\nu\boldsymbol{L}_\nu^{\dagger})\nabla_{\boldsymbol{v}}\mathcal{V}$$

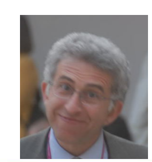
since in general $(\boldsymbol{I} - \boldsymbol{L}_\nu\boldsymbol{L}_\nu^{\dagger})\nabla_{\boldsymbol{v}}\mathcal{V} \neq \boldsymbol{0}$



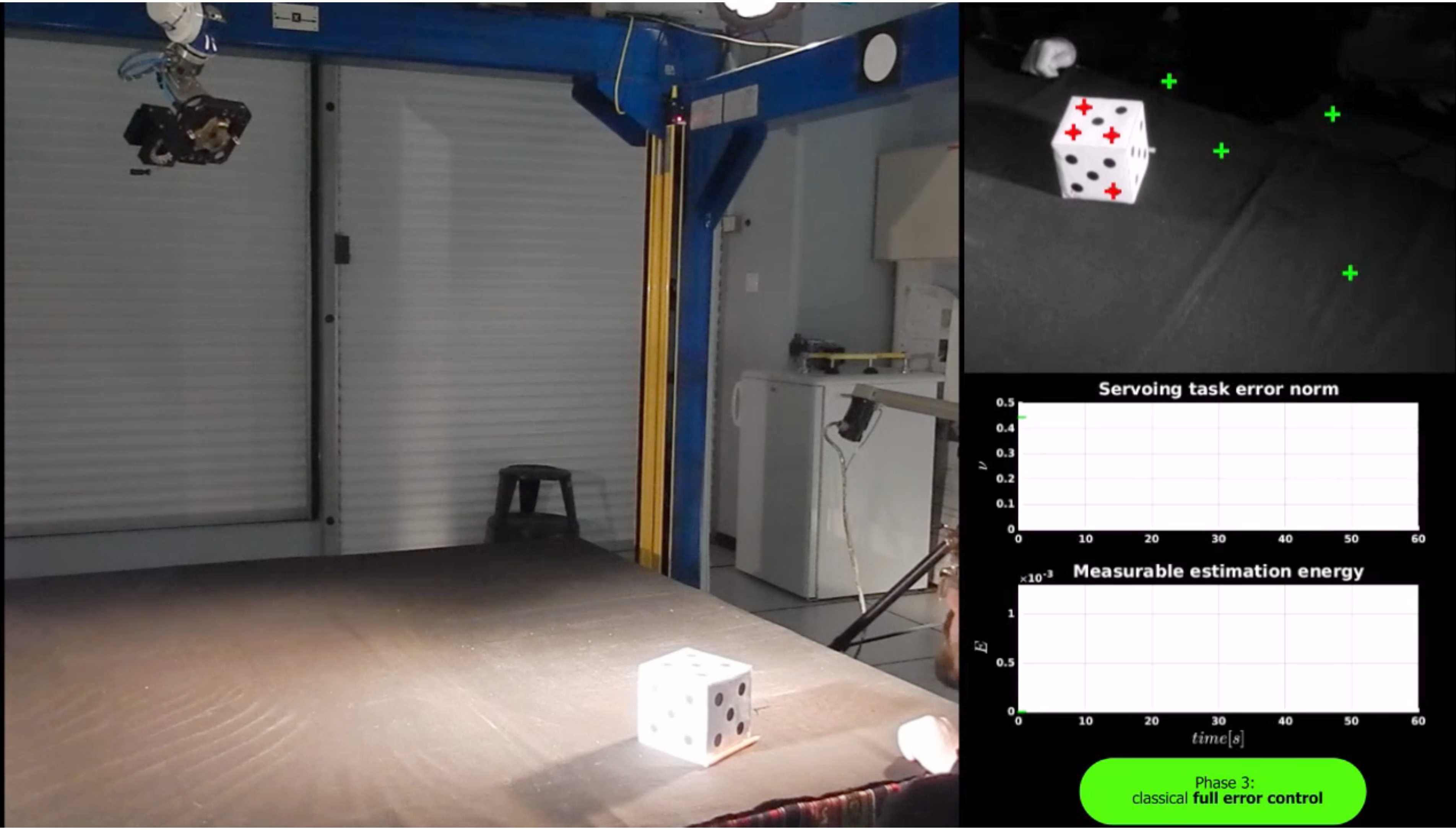


- However, need to switch back to (■) when $\nu \to 0$

- Need of conditions when to switch to (■) and/or re-activate the norm controller
  - Leverage again the explicit expression of the Lyapunov (storage) function



Coupling with Visual Control



Online Optimal Trajectory Planning

- Further developments: all the presented schemes are local/purely reactive

- It would be nicer to optimize over a longer time horizon
  - But in an **online fashion** (for continuously refining the planned trajectory from the estimated state)

- Generic nonlinear dynamics with output noise

$$\dot{\boldsymbol{q}}(t) = \boldsymbol{f}(\boldsymbol{q}(t), \boldsymbol{u}(t)), \quad \boldsymbol{q}(t_0) = \boldsymbol{q_0}$$
$$\boldsymbol{z}(t) = \boldsymbol{h}(\boldsymbol{q}(t)) + \boldsymbol{\nu}$$

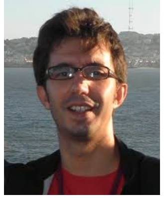
- The Constructibility Gramian $\boldsymbol{\mathcal{G}_c}(t_0, t_f) \triangleq \int_{t_0}^{t_f} \boldsymbol{\Phi}(\tau, t_f)^T \boldsymbol{C}(\tau)^T \boldsymbol{W}(\tau) \boldsymbol{C}(\tau) \boldsymbol{\Phi}(\tau, t_f) \, \mathrm{d}\tau$

$$\dot{\boldsymbol{\Phi}}(t, t_0) = \boldsymbol{A}(t)\boldsymbol{\Phi}(t, t_0), \qquad \boldsymbol{\Phi}(t_0, t_0) = I$$

captures the ability in reconstructing the state $\boldsymbol{q}(t_f)$ at the final time t_f

- One can show that $\boldsymbol{P}^{-1}(t) = \boldsymbol{\Phi}^T(t_0, t)\boldsymbol{P}_0^{-1}\boldsymbol{\Phi}(t_0, t) + \boldsymbol{\mathcal{G}_c}(t_0, t)$

$$\boldsymbol{P}^{-1}(t) = \boldsymbol{\mathcal{G}_c}(-\infty, t)$$


P. Salaris


R. Spica


M. Cognetti

Online Optimal Trajectory Planning

- Natural optimization problem

$$\boldsymbol{u}^*(t) = \arg\max_{\boldsymbol{u}} \ \|\boldsymbol{\mathcal{G}_c}(-\infty, t_f)\|,$$
$$s.t.$$
$$E(t_0, t_f) = \int_{t_0}^{t_f} \sqrt{\boldsymbol{u}(\tau)^T \boldsymbol{M} \boldsymbol{u}(\tau)}\, \mathrm{d}\tau = \bar{E}$$

- However, all quantities depend on $\boldsymbol{q}(t),\ t \in [t_0,\ t_f]$ which is **unknown**

- An offline optimization at $t = t_0$ would be based on $\hat{\boldsymbol{q}}(t_0)$ and thus arbitrarily wrong

- On the other hand, during motion $\hat{\boldsymbol{q}}(t) \to \boldsymbol{q}(t)$ by using an observer (e.g., a EKF)

- Possible solution: continuously refine the optimized path based on the (converging) $\hat{\boldsymbol{q}}(t)$

- Useful decomposition $\boldsymbol{\mathcal{G}}_c(-\infty,\ t_f) = \boldsymbol{\Phi}(t,\ t_f)^T\left(\boldsymbol{\mathcal{G}}_c(-\infty,\ t) + \boldsymbol{\mathcal{G}}_o(t,\ t_f)\right)\boldsymbol{\Phi}(t,\ t_f)$

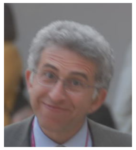
TBO = To Be Optimized

$\boldsymbol{\Phi}(t,\ t_f)^T$ → TBO; $\boldsymbol{\mathcal{G}}_c(-\infty,\ t)$ → Fixed; $\boldsymbol{\mathcal{G}}_o(t,\ t_f)$ → TBO; $\boldsymbol{\Phi}(t,\ t_f)$ → TBO

Online Optimal Trajectory Planning

- Simplifying assumptions

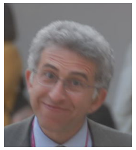
- 1) System $\begin{aligned} \dot{\boldsymbol{q}}(t) &= \boldsymbol{f}(\boldsymbol{q}(t), \boldsymbol{u}(t)), \quad \boldsymbol{q}(t_0) = \boldsymbol{q_0} \\ \boldsymbol{z}(t) &= \boldsymbol{h}(\boldsymbol{q}(t)) + \boldsymbol{\nu} \end{aligned}$ admits a set of flat outputs $\zeta(\boldsymbol{q})$



→ no need to integrate the system dynamics for generating $\hat{\boldsymbol{q}}(\tau),\ \tau \in [t,\ t_f]$ from $\hat{\boldsymbol{q}}(t)$

- 2) the flat outputs $\zeta(\boldsymbol{q})$ are parameterized by a parametric curve (B-Spline) $\gamma(\boldsymbol{x}_c,\ s)$

→ finite-dimensional optimization problem (the control points $\boldsymbol{x}_c$)



- Reformulated optimization Problem

$$\boldsymbol{x}_c^*(t) = \arg\max_{\boldsymbol{x}_c}\ \|\boldsymbol{\Phi}(\boldsymbol{x}_c(t), s_t, s_f)^T \left(\boldsymbol{\mathcal{G}_c}(-\infty, s_t) + +\boldsymbol{\mathcal{G}_o}(\boldsymbol{x}_c, s_t, s_f)\right) \boldsymbol{\Phi}(\boldsymbol{x}_c(t), s_t, s_f)\|_\mu$$

Online Optimal Trajectory Planning

- Additional requirements

$$\boldsymbol{x}_c^*(t) = \arg\max_{\boldsymbol{x}_c} \; \|\boldsymbol{\Phi}(\boldsymbol{x}_c(t), s_t, s_f)^T \left(\boldsymbol{\mathcal{G}_c}(-\infty, s_t) + +\boldsymbol{\mathcal{G}_o}(\boldsymbol{x}_c, s_t, s_f)\right) \boldsymbol{\Phi}(\boldsymbol{x}_c(t), s_t, s_f)\|_\mu$$

$$\begin{aligned}
&1)\ \hat{\boldsymbol{q}}(t) - \boldsymbol{q_\gamma}(\boldsymbol{x_c}(t), s_t) \equiv \boldsymbol{0}\,, \\
&2)\ \mathbf{fl}(\boldsymbol{x}_c(\tau), s_\tau) \neq \boldsymbol{0}\,,\ \forall\ \tau \in [t, t_f] \\
&3)\ E(\boldsymbol{x}_c(t),\, s_t,\, s_f) = \bar{E} - E(s_0,\, s_t)\,,
\end{aligned}$$

where

$$E(s_0,\, s_t) = \int_{s_0}^{s_t} \sqrt{\boldsymbol{u}(\sigma)^T \boldsymbol{M} \boldsymbol{u}(\sigma)}\, \mathrm{d}\sigma$$

- Use of (classical) Task Prioritization for taking into account the several requirements

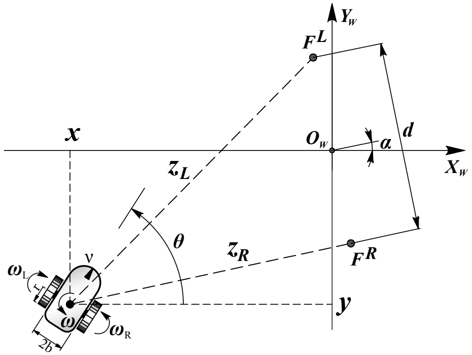
$$\boldsymbol{J}_{A,k} = \begin{pmatrix} \boldsymbol{J}_1^T & \boldsymbol{J}_2^T & \cdots & \boldsymbol{J}_k^T \end{pmatrix}^T \qquad \begin{aligned} \boldsymbol{P}_0 &= \boldsymbol{I} \\ \boldsymbol{P}_k &= \boldsymbol{P}_{k-1} - (\boldsymbol{J}_k \boldsymbol{P}_{k-1})^{\#} \boldsymbol{J}_k \boldsymbol{P}_{k-1} \end{aligned}$$

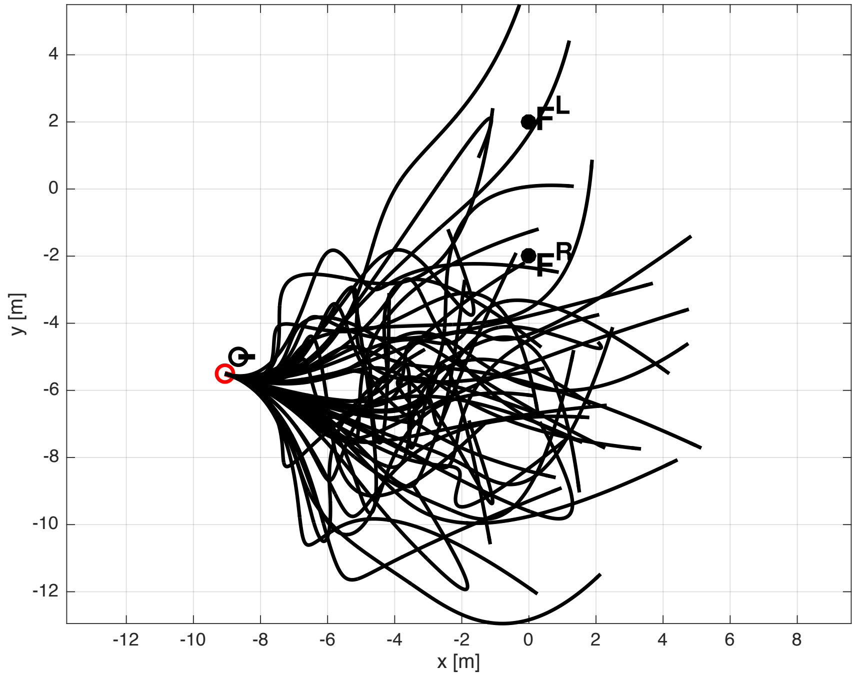
- Control points updated online by following the gradient of the cost in the null-space of the requirements

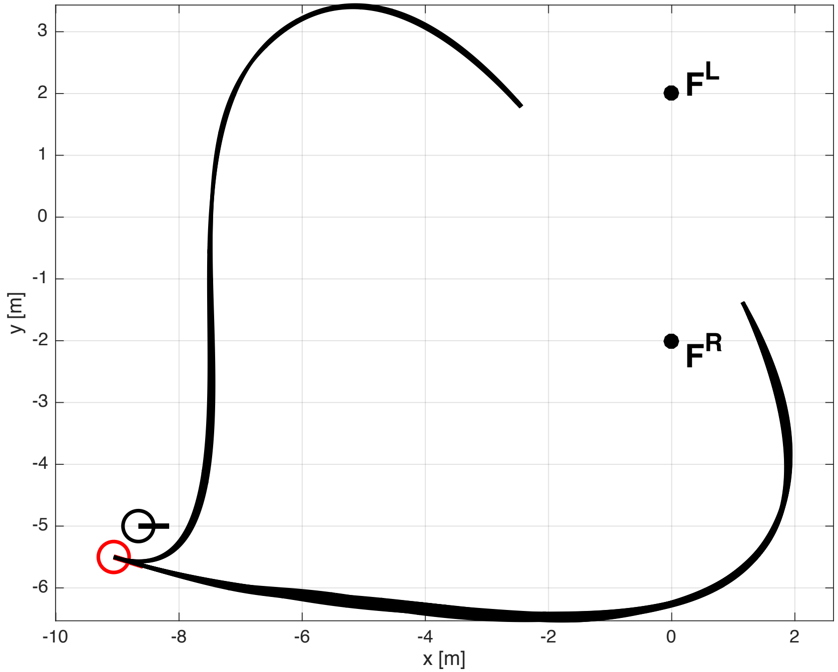
$$\dot{\boldsymbol{x}}_c(t) = \boldsymbol{u}_c(t), \qquad \boldsymbol{x}_c(t_0) = \boldsymbol{x}_{c,0}$$

Online Optimal Trajectory Planning

- Some results for a unicycle measuring two distances

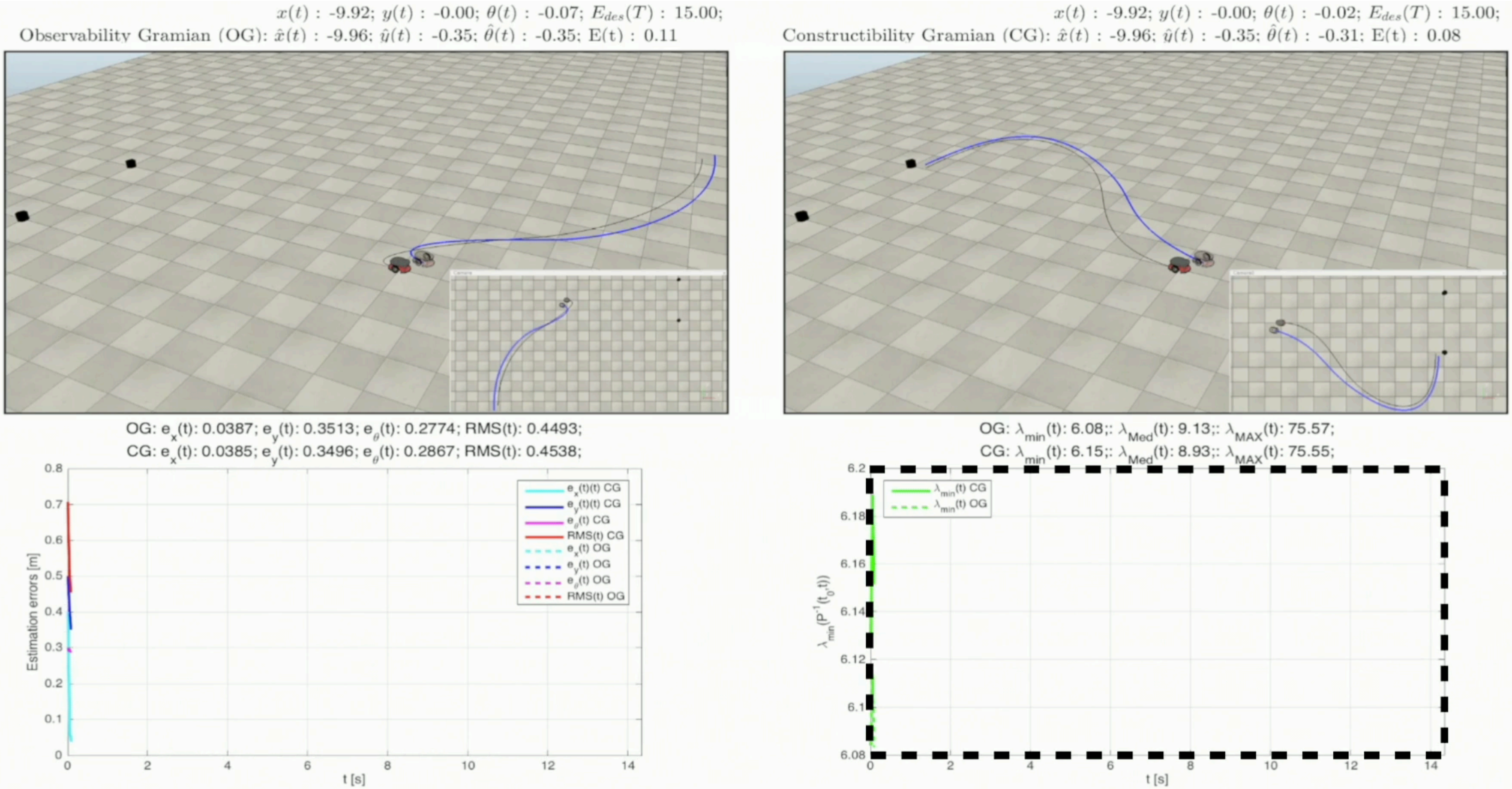






Online Optimal Trajectory Planning

- Some results



Evolution of the smallest eigenvalue of the inverse of the covariance matrix given by the EKF (i.e. of the estimated CG)

Conclusions

- A small selection of how my scientific career was shaped by the interactions with ADL

- Like many others, I chose to work in robotics also inspired by ADL's teaching, passion, mentoring and guidance

- …and he's still a source of inspiration nowadays (in particular his slides and notes ☺)

- I can only thank the ADLipedia for being there !