Learning and Adaptive Control for Robots Course

Structure of the course

Class Format

13h15-15h00

lecture

Live in Class CE 1106
Live *on zoom*

15h15-17h00

Handwritten Exercise session

Live *in class room CE1106*
Live *on zoom & discord*

Matlab Exercise session

Live *in class room CE1106*
Live *on zoom & discord*

4 hours

Class Format

13h15-15h00

lecture

Live in Class CE 1106

Live *on zoom*

! Some courses are in flipped class format – pay attention to schedule!

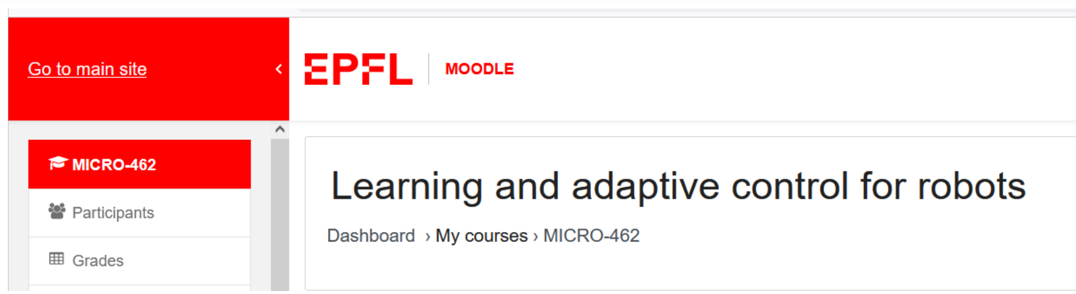
Flipped Class format

Watch videos for 45 minutes

Quiz

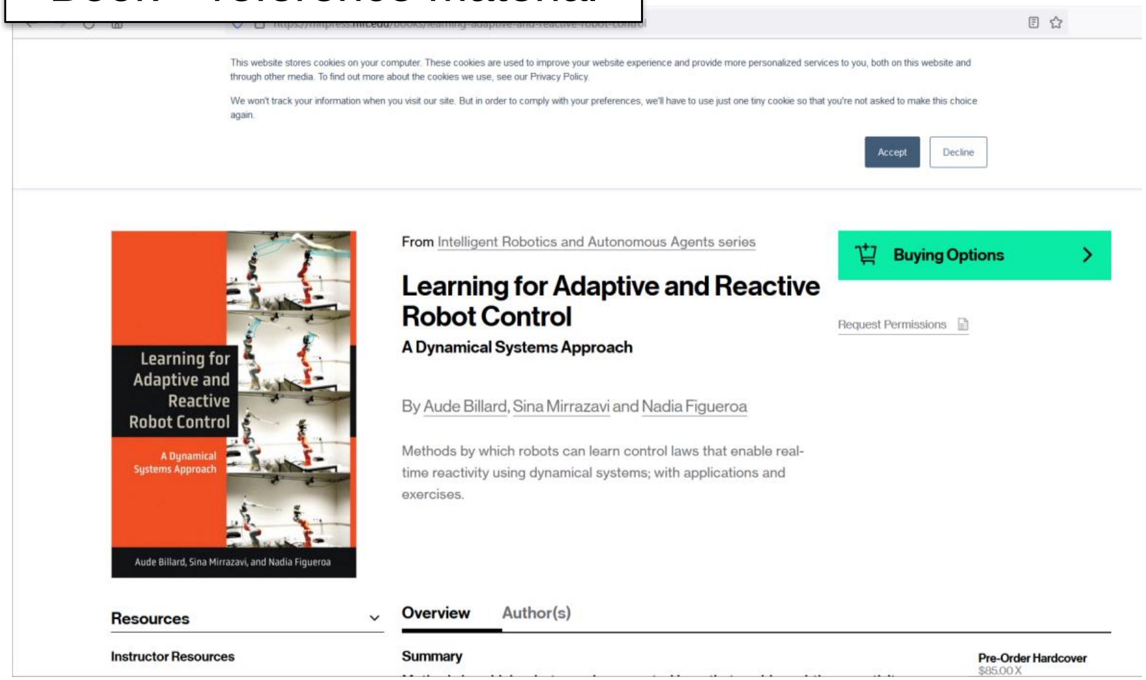
14h15-15h00 Interactive Lecture in class

Material for the class



- Videos
- Slides
- Exercises
- Solutions

Book – reference material



PURCHASE of Hardcopies: the book can be purchased from the EPFL bookstore - Libraire Integrale - with a 10% discount:
ELECTRONIC VERSION: An electronic copy is available for free through the EPFL library via the BEAST Catalog at this LINK.
RENTAL: the book can be rented as e-textbook rentals through https://mitpress.ublish.com/

Grading Scheme

Oral Exam (100)% of the grade

Closed book

Allowed 1 A4 recto-verso page with *handwritten notes*

Software

Matlab 2019 and higher version

Requires following toolboxes:

- statistics and machine learning toolbox
- signal processing toolbox
- robotics system toolbox
- optimization toolbox
- deep learning toolbox
- model predictive control toolbox
- control system toolbox
- curve fitting toolbox

Practice session on robots

- Practice Session 3 will use real robots. It will take place in the robot laboratory of the EPFL LASA laboratory in room ME.A3.455.

- Practice session must be done **by team of two**.

- We have 5 sets of 2-hours long sessions on
  - May 12: 1-3pm or 3-5pm,
  - May 19 3-5pm
  - June 2: 2-4pm or 4-6pm

- **Register for one of the sessions** on moodle by March 4. Past this deadline, we will assign randomly students for this session.

Introduction

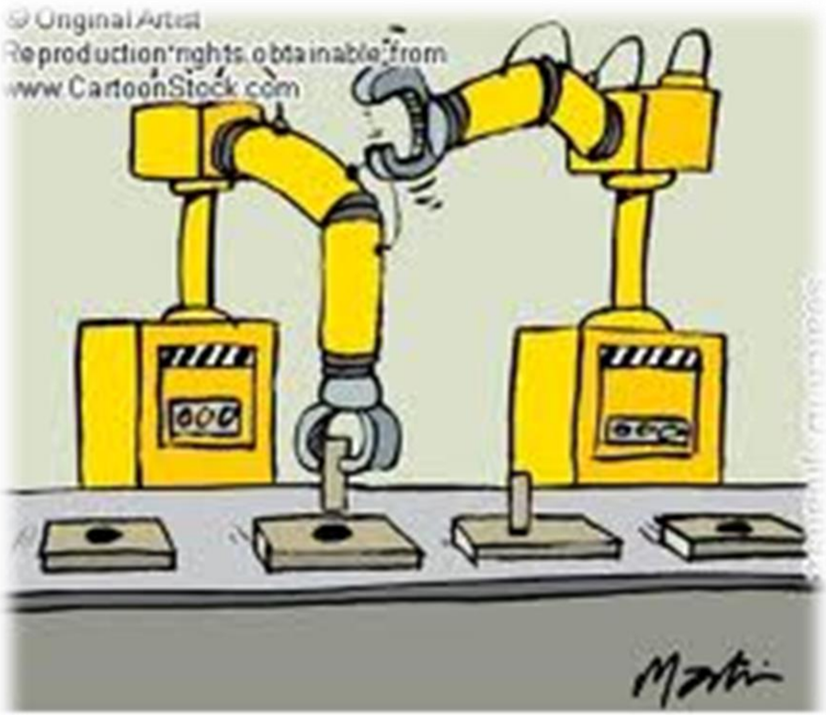
Planning in Robotics

Planning with Dynamical Systems

Outline of the course' material

Traditional Robot Factories

A world fully predetermined, where there is no room for change.



YOU KNOW, IF IT WASN'T FOR THE BORING REPETITION, THIS JOB WOULD BE THE PITS!

Traditional Robot Factories

Robots’ motions are **preprogrammed**,
always the same, optimized for maximum efficiency

Traditional Robot Factories

A world without humans

Industry 4.0

- Robots can work outside cells and work collaboratively with humans
  → this may increase productivity and save space.
- New standards: ISO 10218, ANSI/RIA R15.06

Robot cell

Sources: Iris Electronics, Fraunhofer IPA

Human-Robot Co-workers

EPFL / LASA

Needs for real-time planning

Commercial airplanes are already to a large extent driven autonomously

But the environment is only partially predictable

→ need to learn from data

→ need guarantees on stability of the learned controller

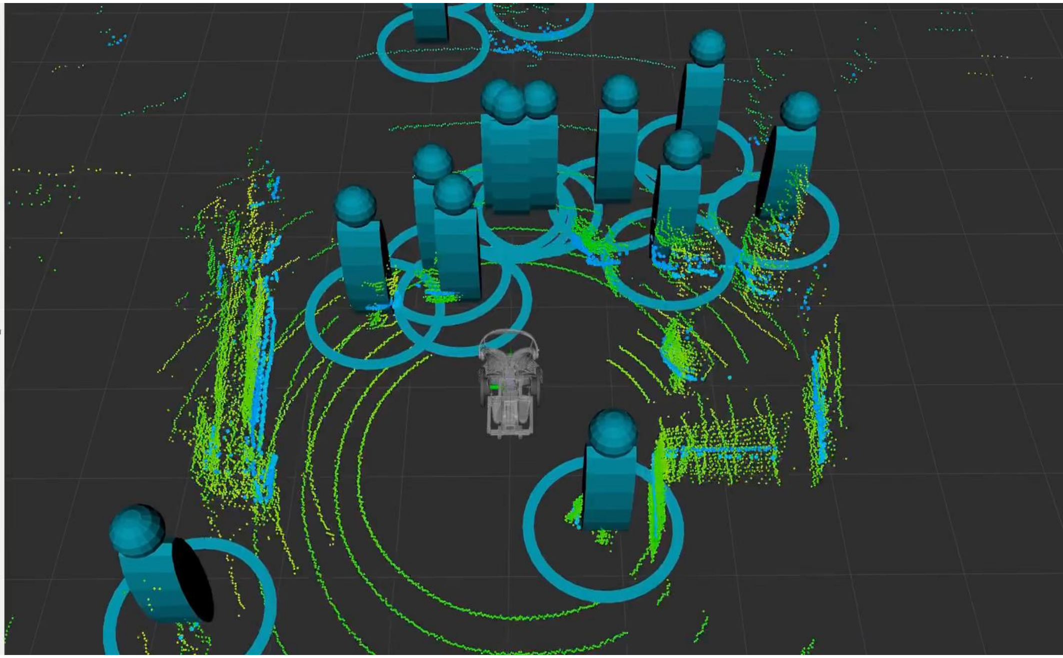
Needs for real-time planning

Autonomous mobility device and wheelchairs will soon navigate our streets.

They must remain safe for both their users and bystanders.

Needs for real-time planning

The environment is only partially observable.

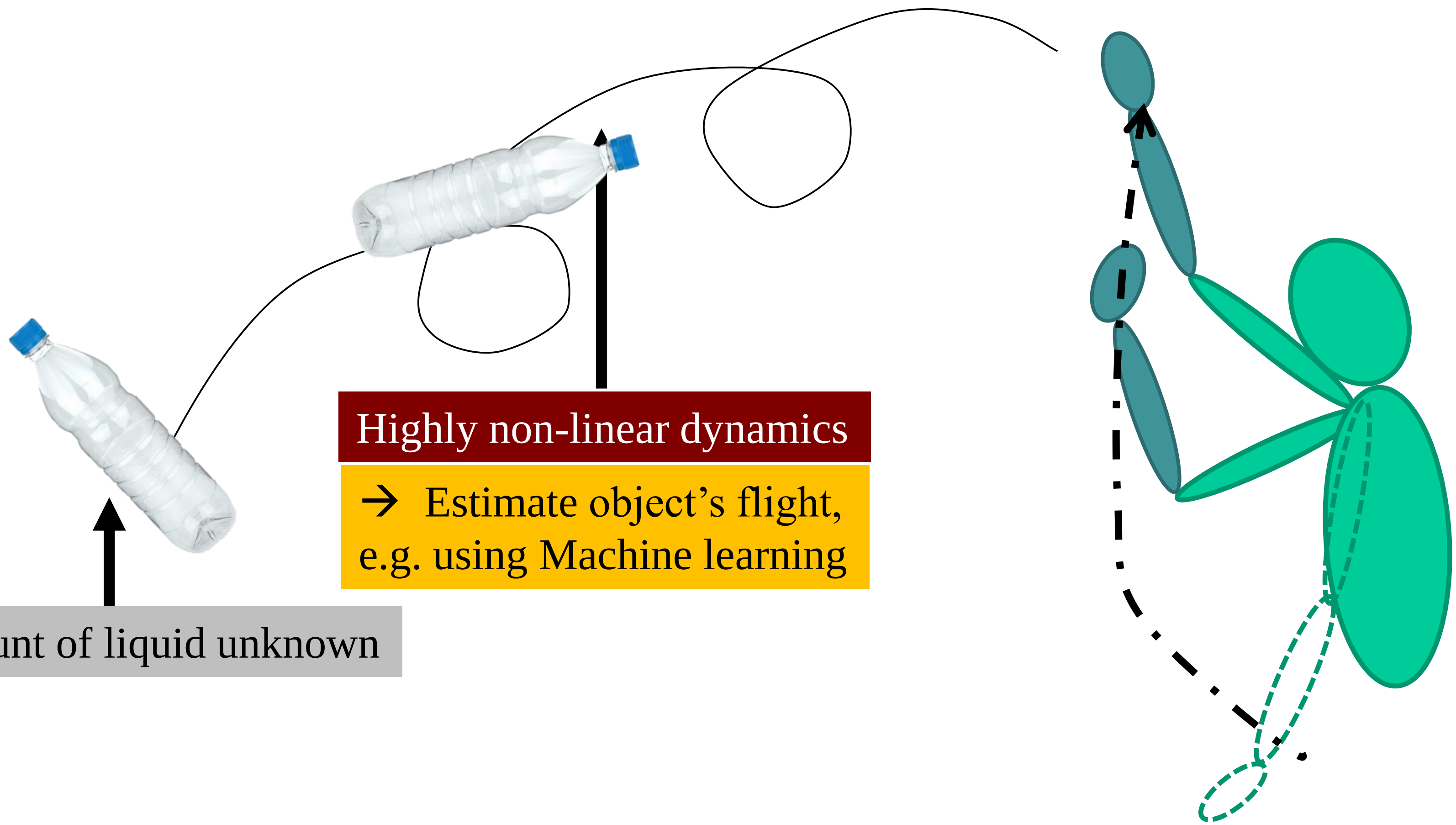


There is a need to react in milliseconds to avoid collisions.

Truly real-time planning

Challenges to real-time planning

Time of flight ~ 0.4sec. → Need to start moving right away



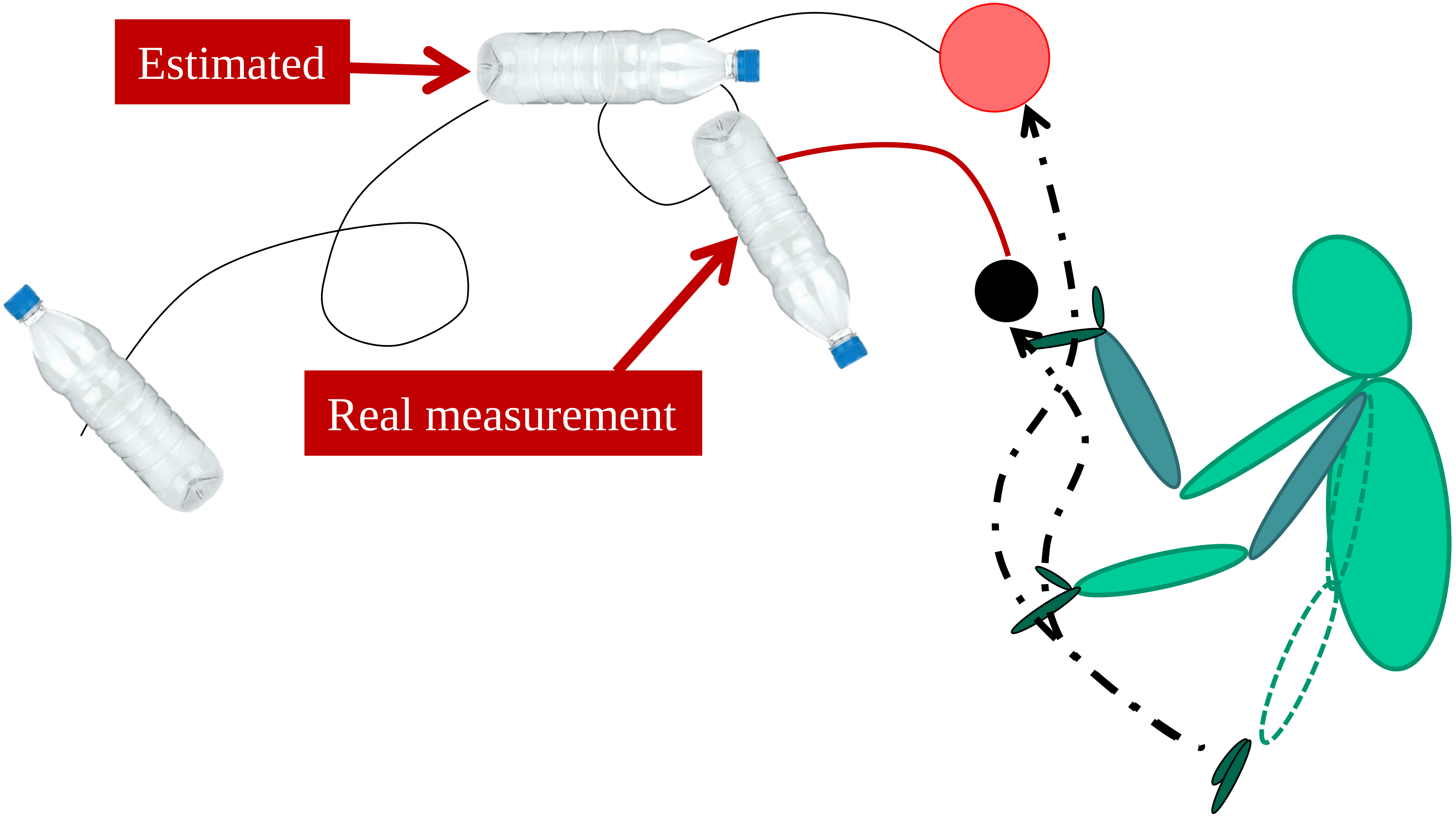
Highly non-linear dynamics

→ Estimate object's flight, e.g. using Machine learning

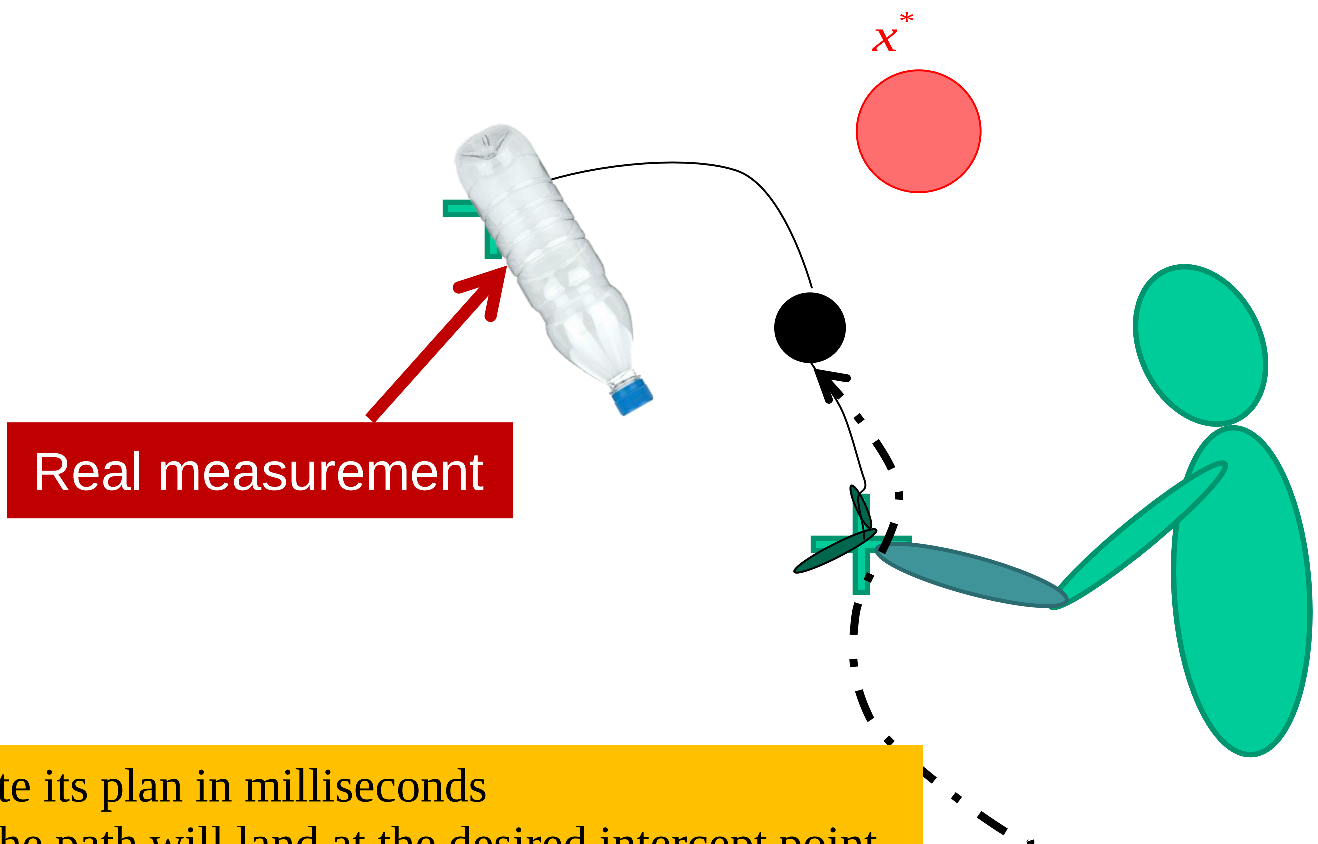
Amount of liquid unknown

Challenges to real-time planning

Inaccurate / frame drop



Challenges to real-time planning



→ The planner must update its plan in milliseconds
→ It must guarantee that the path will land at the desired intercept point.

Challenges faced by real-time planners

- Environment is dynamic
- Environment is only partially predictable
- Environment is only partially observable

- Model of the environment cannot always be explicitly described by known equations.
- One must learn appropriate dynamics for the robot to move in the environment.
- The learned controller must offer guarantees to ensure safety of users and bystanders, and to ensure successful task completion.

Traditional Planning Approaches in Robotics

Path planning in 2D

Path planning, also known as motion planning, was for a long time thought of the problem to move a vehicle (wheel-based) in a 2D environment.

Complexity of the planning relate, primarily, to three factors:

- The vehicle is holonomic or non-holonomic

Path planning in 2D

Path planning, also known as motion planning, was for a long time thought of the problem to move a vehicle (wheel-based) in a 2D environment.

Complexity of the planning relate, primarily, to three factors:

- The vehicle is holonomic or non-holonomic
- The environment is fully or partially known
- The environment is deterministic or stochastic

Overview of Path Planning Approaches

Global Path Planning

- Compute all paths – complete search
- Determine the set of path that are optimal

Pros:
Guarantees:
- optimality
- completeness of the search
- convergence to the goal
- feasibility of the paths

Cons:
Requires complete enumeration
Depends on global knowledge of the world
Does not apply to continuous world representations

Local Path Planning

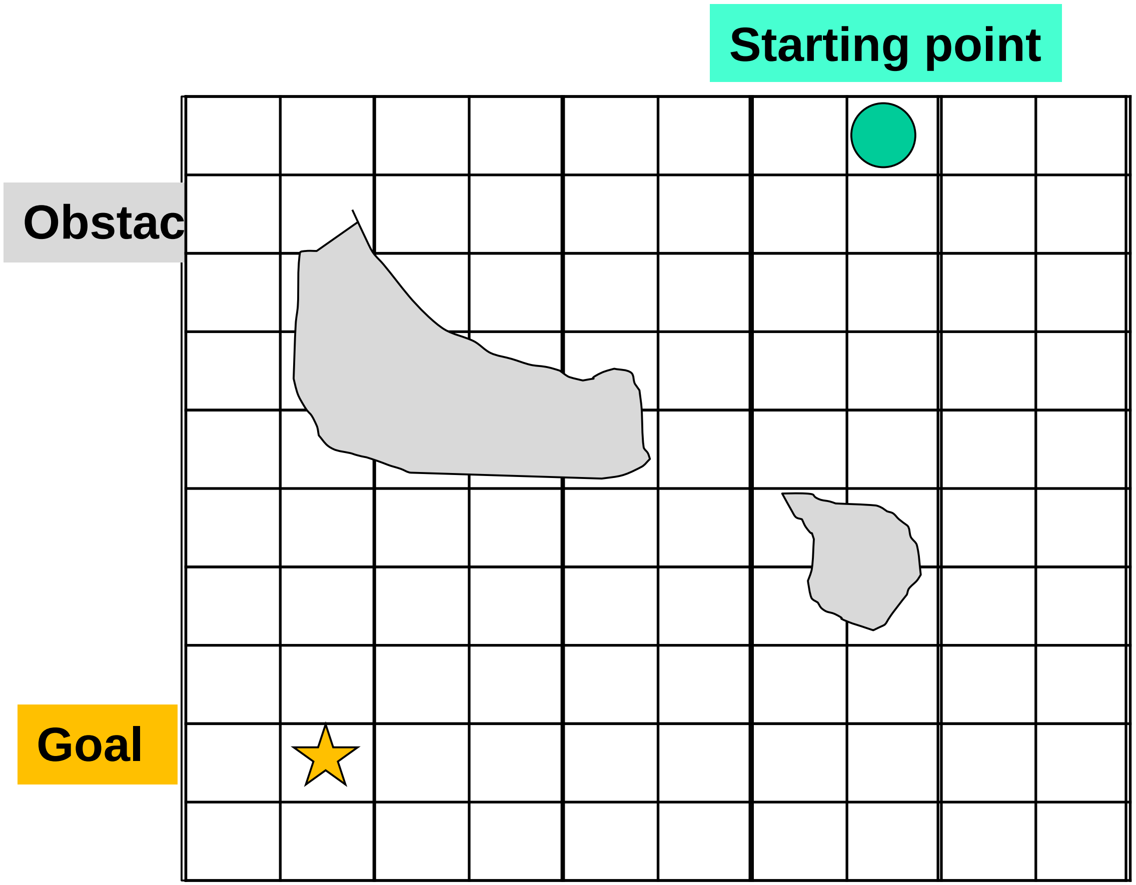
- Compute a subset of paths in a neighborhood
- Determine the optimal paths among this set

Pros:
Requires only local knowledge of the world

Guarantees:
- fast and reactive control
- adapted to real-time control
- adapted to local robot perception

Cons:
May not find a feasible path to the goal
Paths may all be suboptimal
Depends on a heuristics to determine the paths

Global Path Planning Approach – Discrete case



Global navigation

Cell decomposition:
→ Discretization of the actions and states

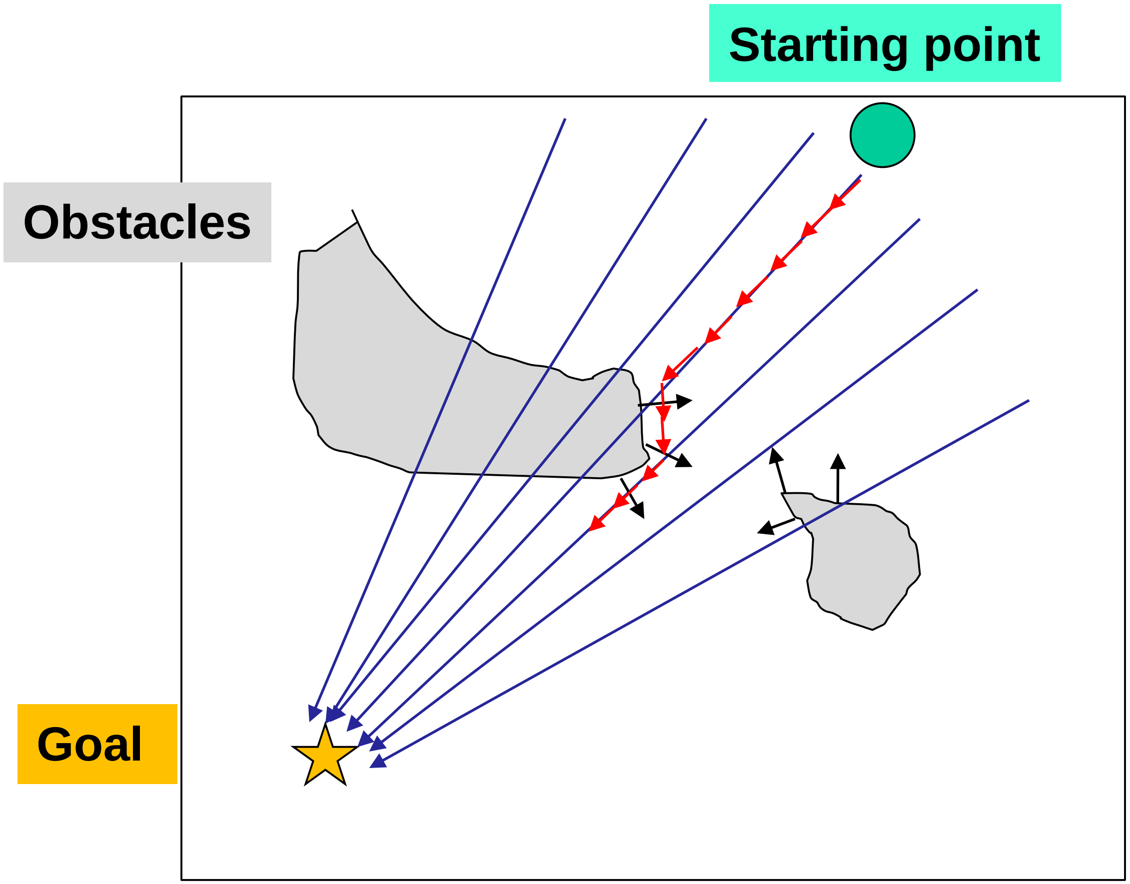
To find a optimal path that is devoid of obstacles depends on how fine the granularity of the decomposition is.

Computing all paths becomes quickly intractable as the size of the world increases

Having discrete states leads to jerky motions.

EPFL

Global Path Planning Approach – Continuous case



Global navigation

Vector field dragging towards the goal
→ Potential of energy

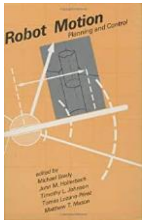
Vector field repulsing from obstacles

Final motion – combination of the two

First approaches by O. Khatib suffered from local minima and required fine tuning of combination.
Other approaches by J-J Slotine, and D.E. Koditschek offer theoretical guarantees with no or few minima.

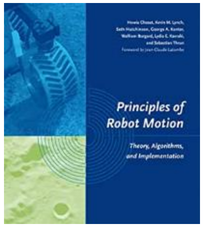
History of Robot Motion Planning through Books

- Robot Motion: Planning and Control
 Michael Brady, John Hollerbach, Tomá s Lozano-Pé rez, Matthew Mason
 MIT Press, 1983

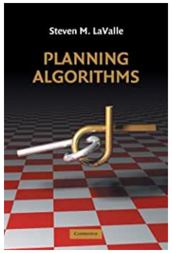


- Robot Motion Planning,
Jean-Claude Latombe,
Kluwer Academic Publishers, Boston, MA, 1991.

- Principles of Robot Motion: Theory, Algorithms, and Implementations,
H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard,
L. E. Kavraki and S. Thrun,
MIT Press, Boston, 2005.



- Planning Algorithms,
Steven M. LaValle,
Cambridge University Press, May 29, 2006. http://planning.cs.uiuc.edu



- Robot Motion Planning and Control,
Jean-Paul Laumond,
Lectures Notes in Control and Information Sciences, 2009.

- Motion Planning for Humanoid Robots,
Kensuke Harade, Eiichi Yoshida, Kazuhito Yokoi,
Springer-verlag, 2010

Partial History / Recent works on Potential Field Methods

Foundations:

- Khatib, Oussama. "The potential field approach and operational space formulation in robot control." *Adaptive and learning systems*. Springer, Boston, MA, 1986. 367-377.
- Khatib, Oussama. "Real-time obstacle avoidance for manipulators and mobile robots." *Autonomous robot vehicles*. Springer, New York, NY, 1986. 396-404.
- Koditschek, Daniel. "Exact robot navigation by means of potential functions: Some topological considerations." *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. IEEE, 1987.
- Rimon, Elon. "Exact robot navigation using artificial potential functions." PhD diss., Yale University, 1990.
- Feder, Hans Jacob S., and J-JE Slotine. "Real-time path planning using harmonic potentials in dynamic environments." *Proceedings of International Conference on Robotics and Automation*. Vol. 1. IEEE, 1997.

More recent works:

- Khansari-Zadeh, Seyed Mohammad, and Aude Billard. "A dynamical system approach to realtime obstacle avoidance." *Autonomous Robots* 32, no. 4 (2012): 433-454.
- Arslan, Omur, and Daniel E. Koditschek. "Sensor-based reactive navigation in unknown convex sphere worlds." *The International Journal of Robotics Research* 38.2-3 (2019): 196-223
- Huber, Lukas, Aude Billard, and Jean-Jacques Slotine. "Avoidance of convex and concave obstacles with convergence ensured through contraction." *IEEE Robotics and Automation Letters* 4.2 (2019): 1462-1469.
- Loizou, Savvas, and Elon Rimon. "Mobile Robot Navigation Functions Tuned by Sensor Readings in Partially Known Environments." *IEEE Robotics and Automation Letters* (2022).

Path Planning with Learned Dynamical Systems

Path planning with Dynamical systems:

- Generalization of potential field methods.
- Offers closed-form / analytical description of all feasible paths.
- Can be combined with machine learning to learn the vector field.
- Generates non-linear dynamics for the robot's paths with stability guarantees.

Advantages:

Combines advantages of global planning techniques and of local planning techniques.

- As global planning techniques, guarantees convergence to the goal.
- As local planning techniques, ensures fast and reactive control.

Limitations :

- Requires knowledge of the world (location of goal, location of obstacles).
- Depends on having a set of examples of feasible paths to learn from.
- Learned path optimal only if demonstrated paths are optimal.
- Accuracy of models of the path depends on accuracy of the machine learning technique.

Motion planning for an articulated robot arm

Path planning for controlling robot arms

The dimension of the control space for a robot arm is much larger than that of a vehicle moving in 2D

- Usually a robot arm has between 4 to 7 degrees of freedom
- A robot hand has between 1 (gripper) to 22 degrees of freedom (anthropomorphic hands)

4 DOFs arm manipulator + 1 DOF gripper used in the matlab simulations of this class.

Franka Emika

7 DOFs arm manipulator + 1 DOF gripper used in practice sessions of the course

In this course, we assume a fully controllable system: # joints = # actuators

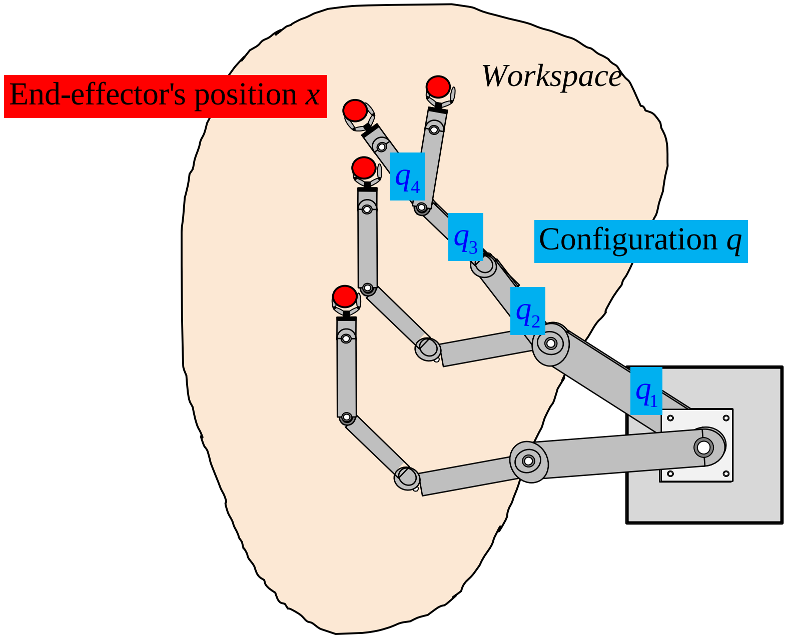
Configuration space versus task space

- The feasible space of motion of the *joints* is called the *configuration space*.
- The feasible space of motion of the robot's end-effector in Cartesian space is called the *workspace*.

Configuration space: C: $\left\{ q \in \mathbb{R}^{N_q} ; q_i^{\min} \leq q_i \leq q_i^{\max} \right\}$

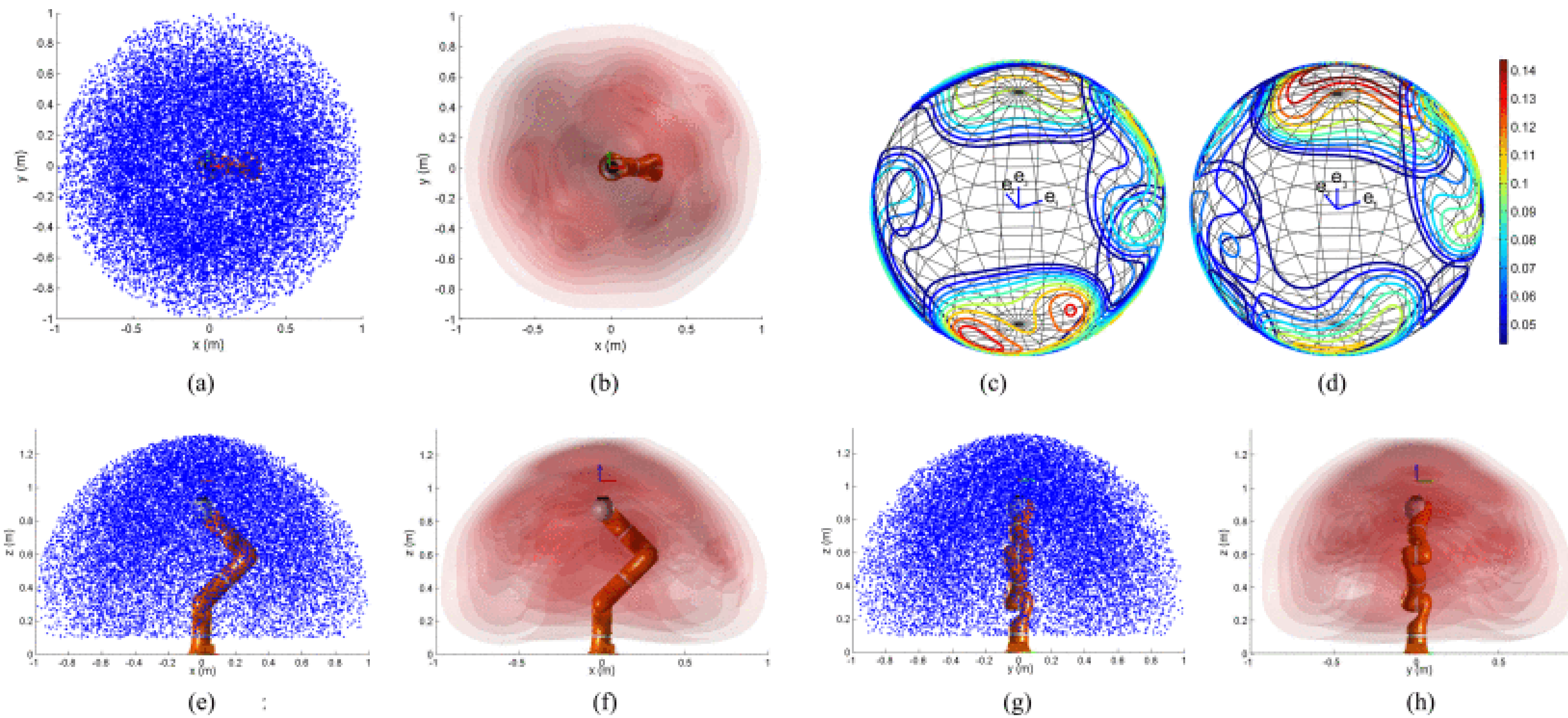
Workspace: W: $\left\{ x \in \mathbb{R}^{N_x} ; \exists q, \text{ s.t. } x = h(q) \right\}$: h: forward kinematics

Usually, N_q =7 and N_x =3 or N_x = 6.



Learning a Model of the Configuration space

The configuration space, or C-space, of the robot system is the space of all possible configurations of the system.



For a robot whose kinematic chain is known, one can sample the space and learn a model of the configuration space, as a distribution of joint configuration. Above: a model for the 7 DOFs KUKA LWR arm has been learned using Gaussian Mixture Model.

Kim et al." IEEE Transactions on Robotics 30, no. 5 (2014): 1049-1065.

Configuration space versus task space

- We control the robot's joints in *joint space*.
- But usually the task constraints are expressed in Cartesian space.

→ To ensure that commands sent in joint space correspond to the desired path in task space, one uses an *Inverse Kinematics* solver.

Given a joint configuration q, there is a unique associated configuration of the end-effector x, given by the Forward Kinematics: $x = h(q)$

Conversely, given a configuration of the end-effector x, there is one or several associated joint configurations, given by the Inverse Kinematics: $q = h^{-1}(x)$.

The inverse is unique solely when the dimension of the joint space equals that of the Cartesian space, e.g. with a 2D planar robot arm moving in a plane.

The damped least-squares inverse kinematics is applied on the velocity flows and is given by:

$$\dot{q} = J^T(q)\left(J(q)J^T(q) + \lambda I\right)^{-1}\dot{x}$$

$\dot{x}$ is the desired velocity of the end-effector,

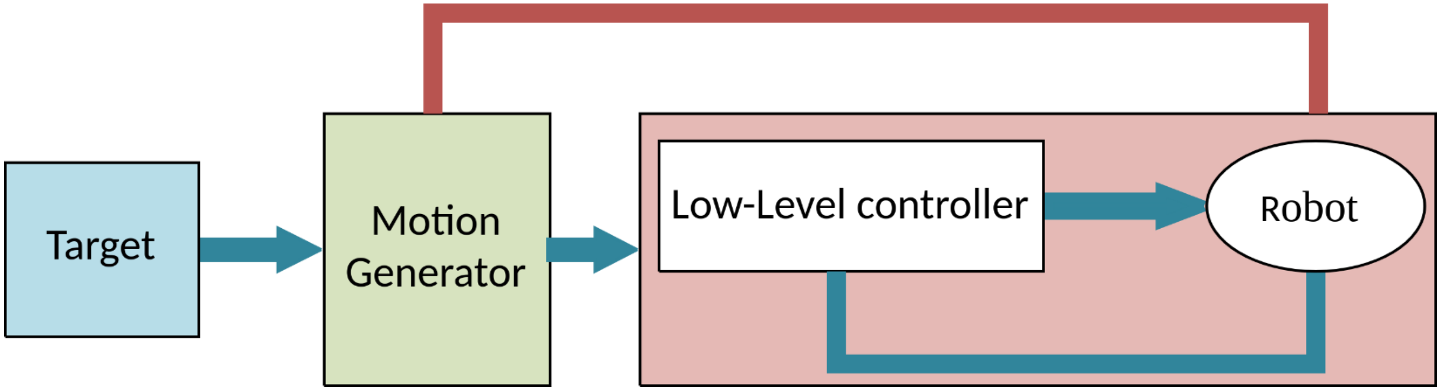
$\dot{q}$ is the associated desired velocity for the joints.

Particularly suited for this course, since the control methods of this class will generate a desired velocity command.

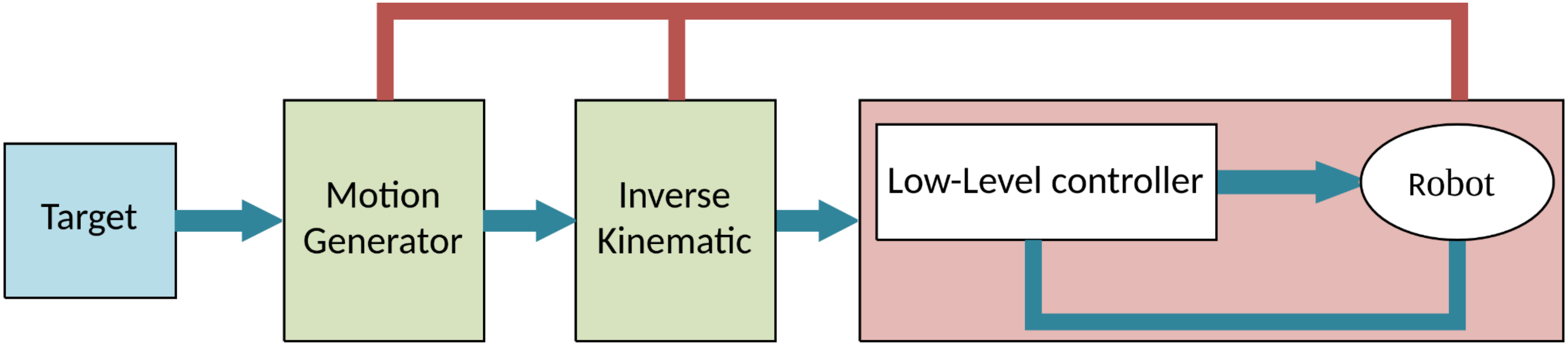
While the examples we will see in this course will be in 2D Cartesian space (task space), most of the algorithms can be applied to higher dimensions. In several of the robotic examples, we will see applications to control directly in joint space.

Control loop in this course

Joint-based control



Cartesian-based control



Path Planning using Optimal Control

Planning a path

$x \in \mathbb{R}^3$: Path in Cartesian space

$\theta \in \mathbb{R}^N$: Joint angles

Solution 1: Optimal control

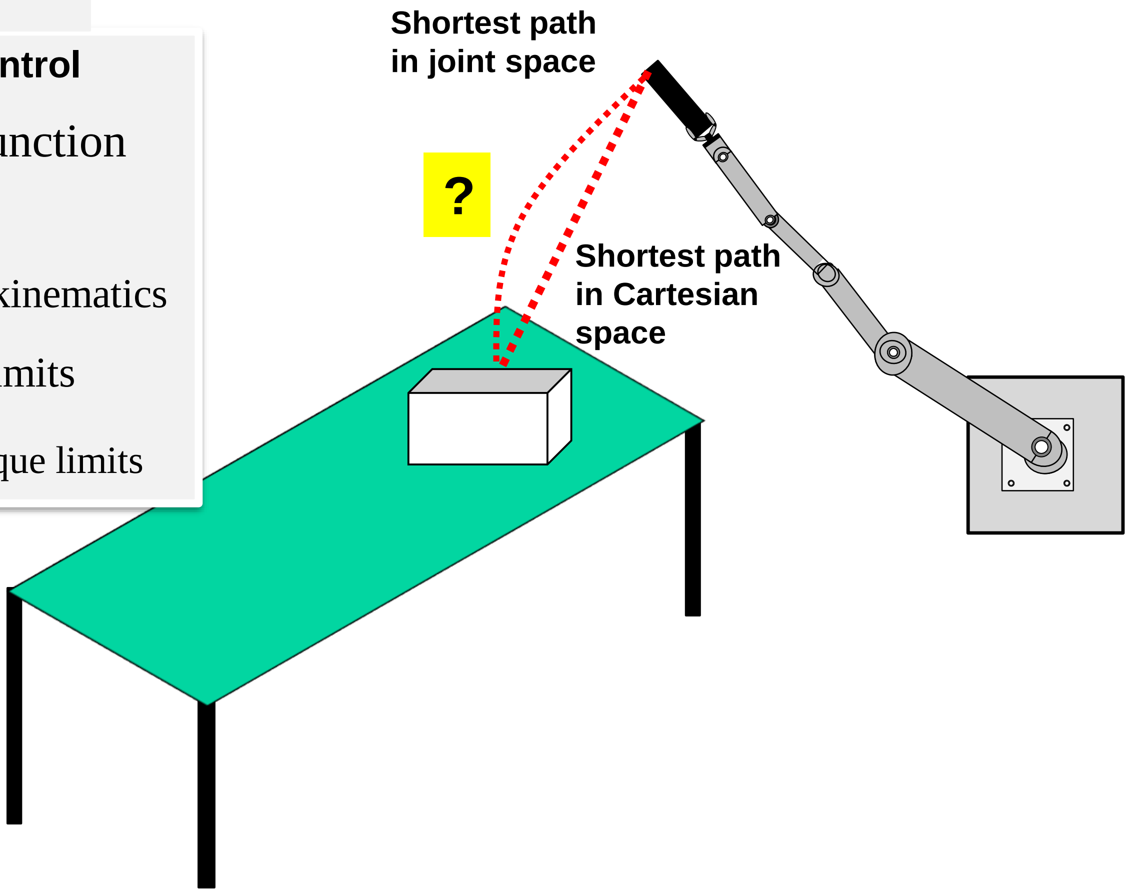
$$\min_{\theta} J(x(\theta)): \text{ cost function}$$

under constraints:

$\dot{x} = J(\theta)\dot{\theta}$ Forward kinematics

$\theta_{\min} \leq \theta \leq \theta_{\max}$ Joint limits

$\ddot{\theta}_{\min} \leq \ddot{\theta} \leq \ddot{\theta}_{\max}$ Joint torque limits



Planning a path with an obstacle

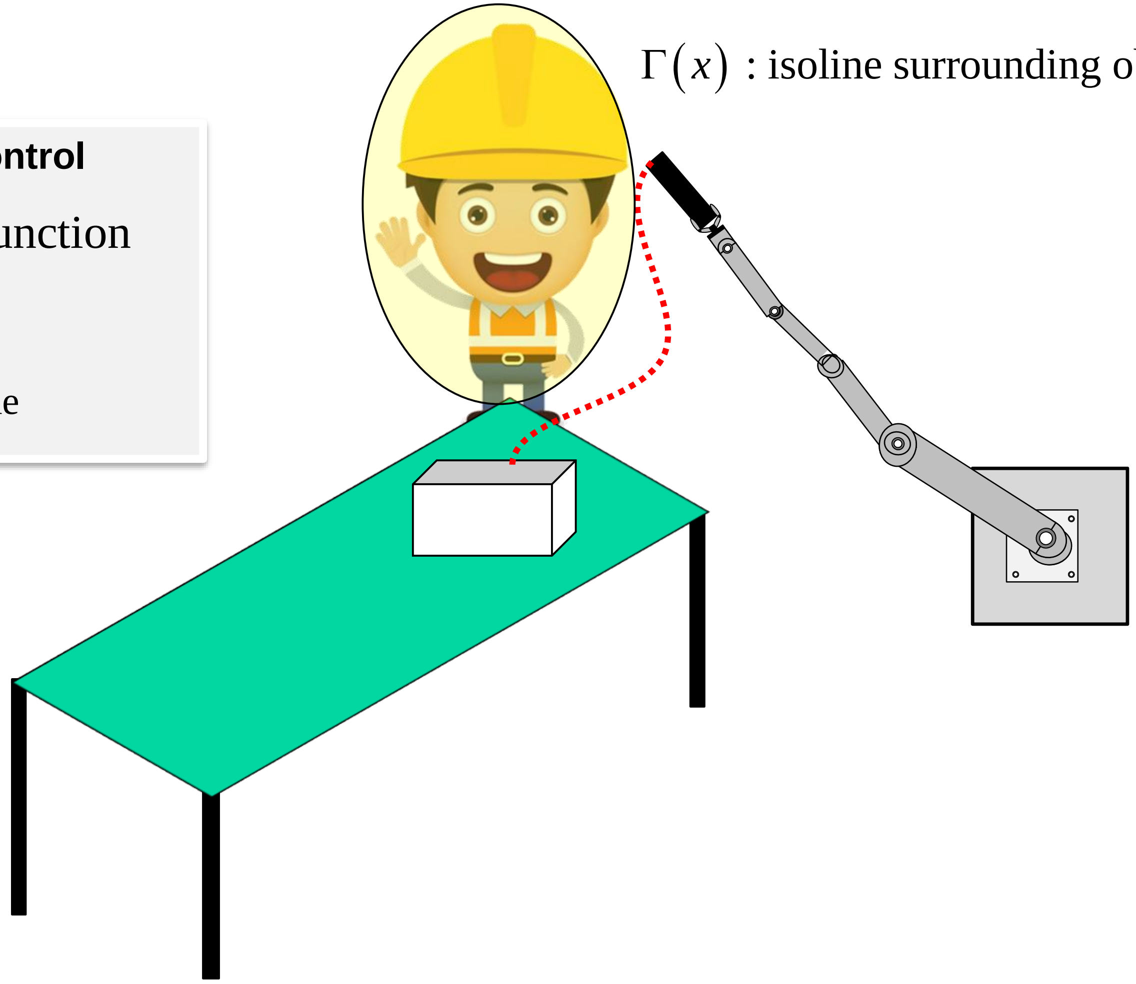
Solution 1: Optimal control

$\min_{\theta} J(x(\theta))$: cost function

under constraints:

$\Gamma(x) \geq x$ Avoid obstacle

$\Gamma(x)$: isoline surrounding obstacle



Planning a path with an obstacle

Discretize the path

Solution 1: Optimal control

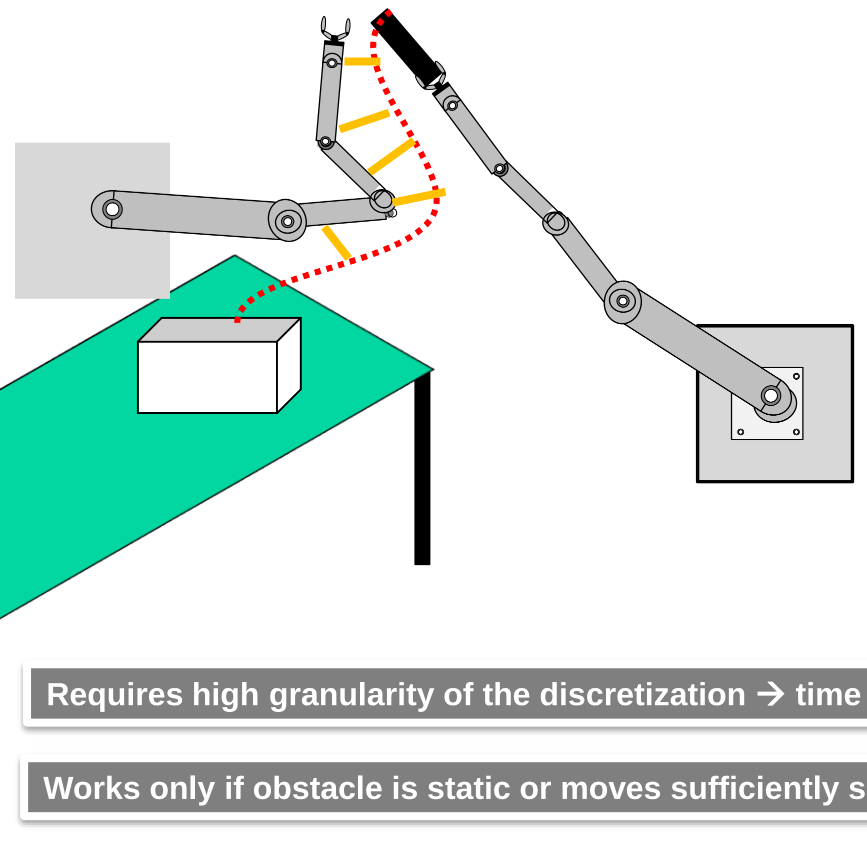
$$\min_{\theta} J(x(\theta)) = \sum_{t=1}^{T} x_i(\theta_i)$$

under constraints:

$\Gamma(x) \geq x$ Avoid obstacle

No closed-form solution

Compute distance at each segment of path



Requires high granularity of the discretization → time consuming

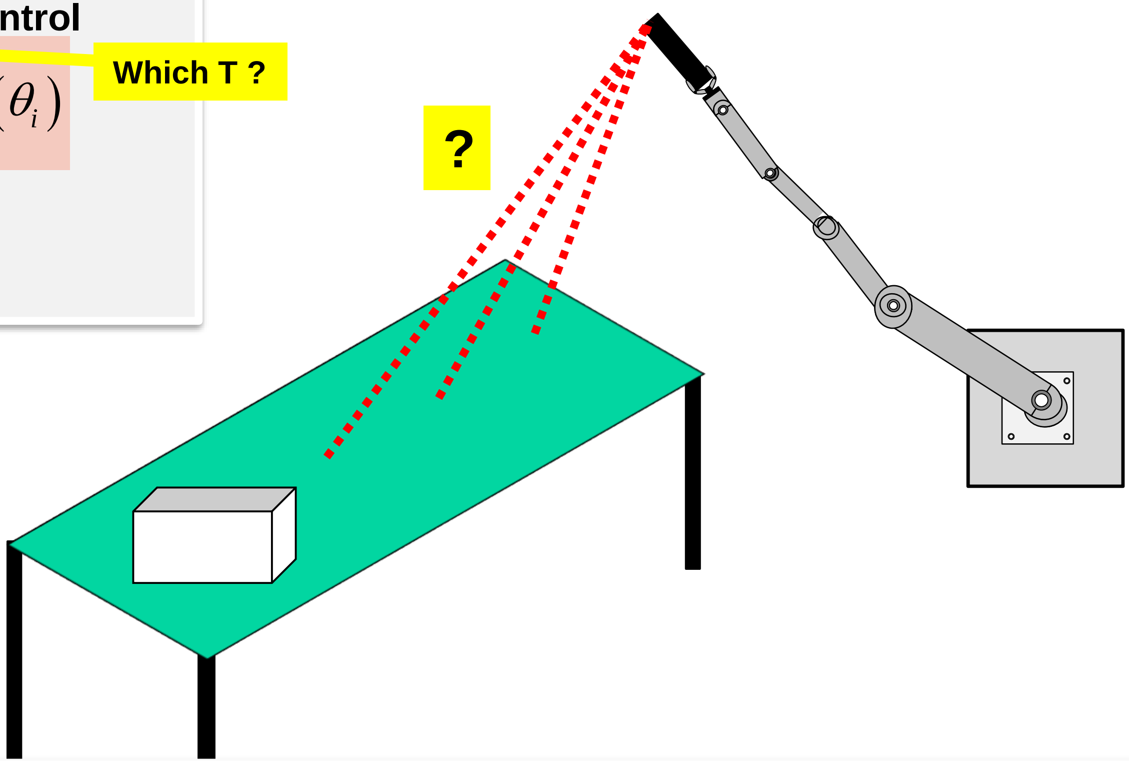
Works only if obstacle is static or moves sufficiently slowly.

Adapting the path when the target moves

Solution 1: Optimal control

$$\min_{\theta} J(x(\theta)) = \sum_{t=1}^{T} x_i(\theta_i)$$

Which T ?



- Each time the environment changes, planning must be redone, which is time consuming.
- Optimal control planners require to fix the time horizon.
- It is not always easy to determine the right time horizon.

Path Planning using Dynamical Systems

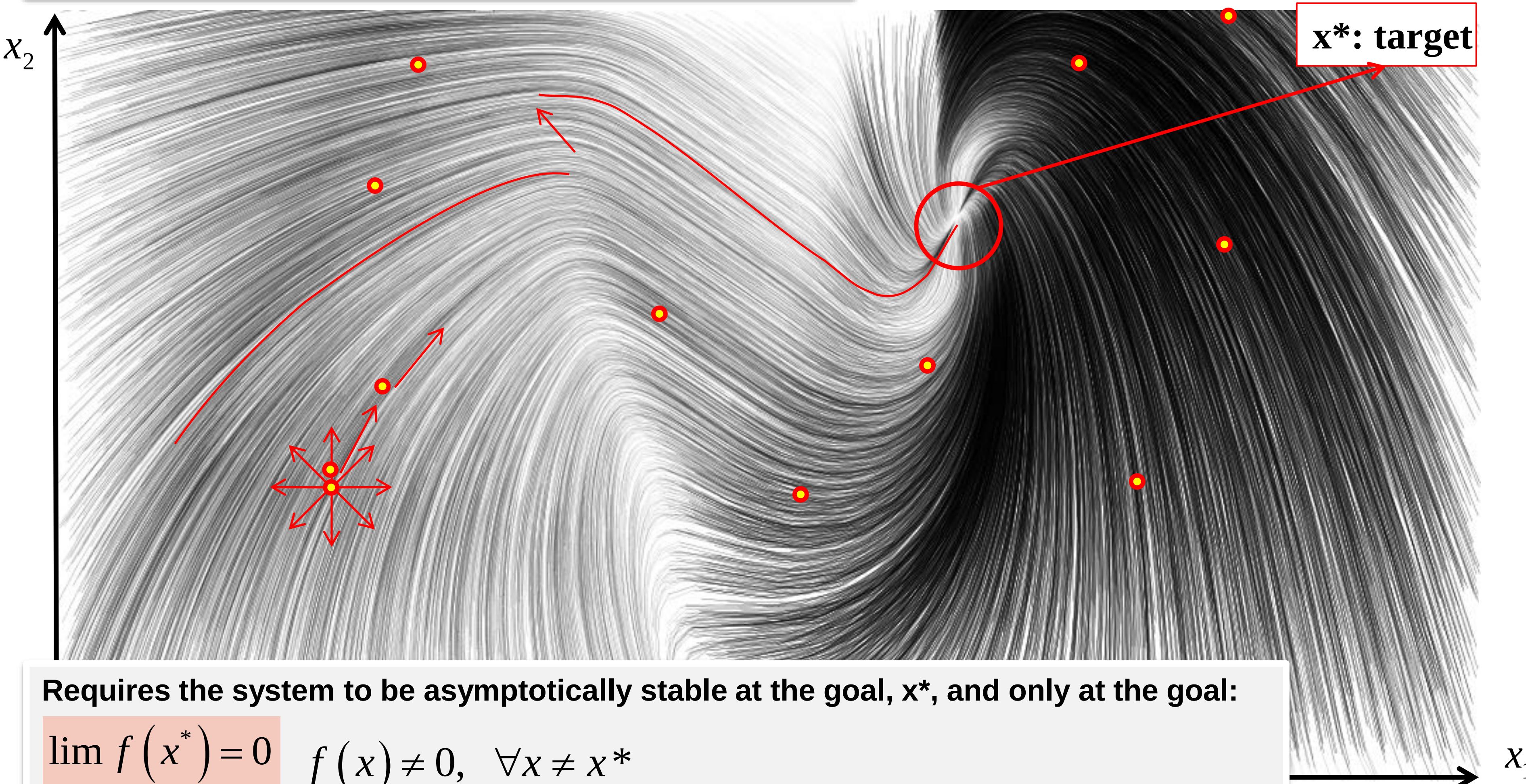
Path planning with Dynamical Systems (DS)

DS control law (1st order ordinary differential equation)

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

$x \in \mathbb{R}^2$: Path in Cartesian space

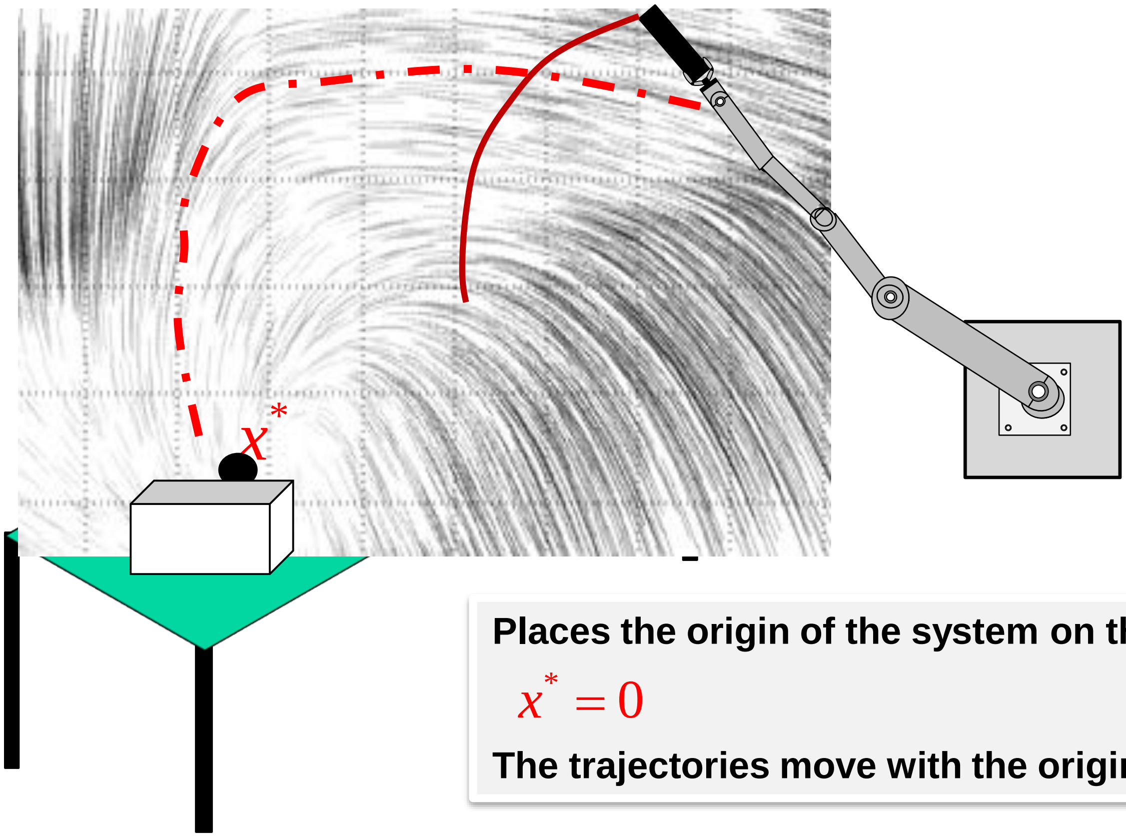
$\dot{x} \in \mathbb{R}^2$: Time-derivative of state, velocity



Requires the system to be asymptotically stable at the goal, x*, and only at the goal:

$$\lim_{t\to\infty} f\left(x^*\right) = 0 \qquad f(x) \neq 0, \quad \forall x \neq x^*$$

Robustness to change in location of the target

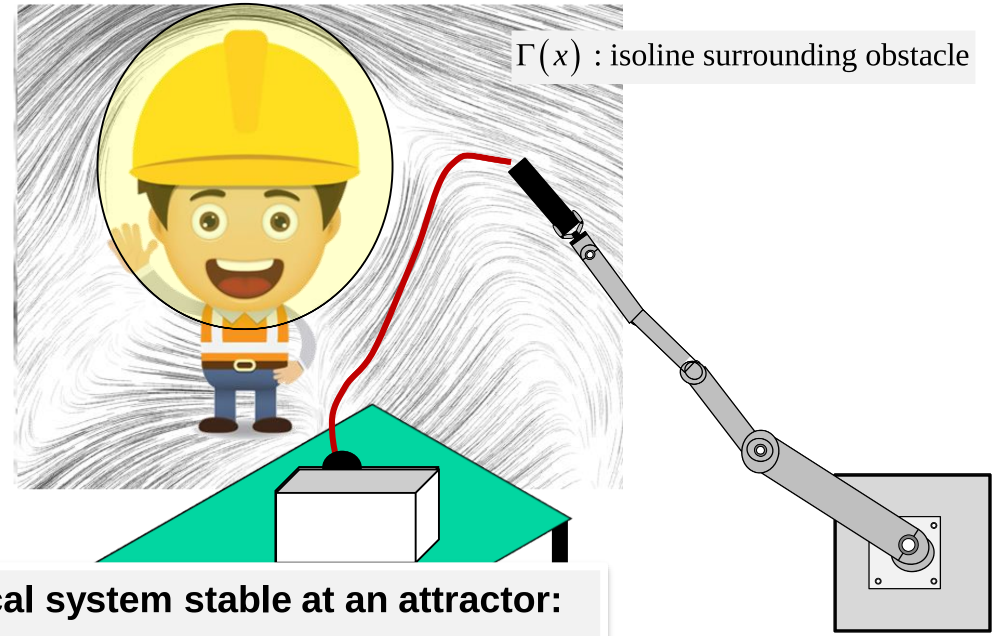


Places the origin of the system on the attractor

$$x^* = 0$$

The trajectories move with the origin

Obstacle avoidance with DS



Starts with an initial dynamical system stable at an attractor:

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

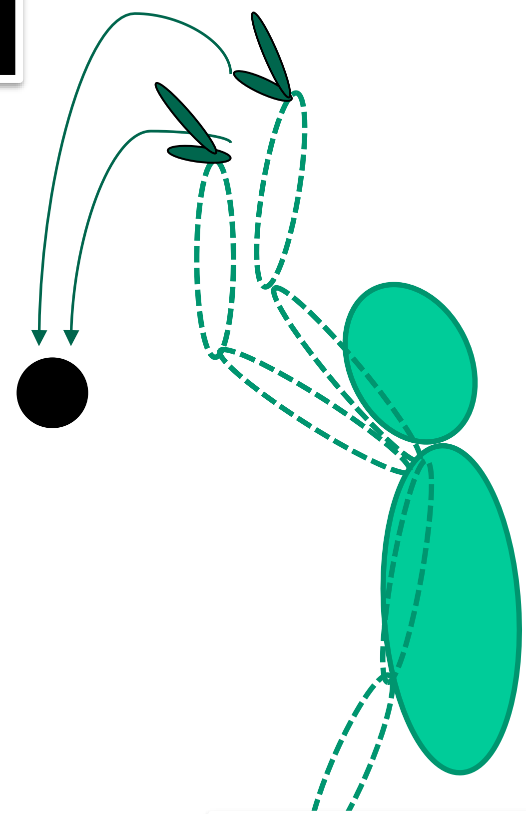
Add a modulation around the obstacle:

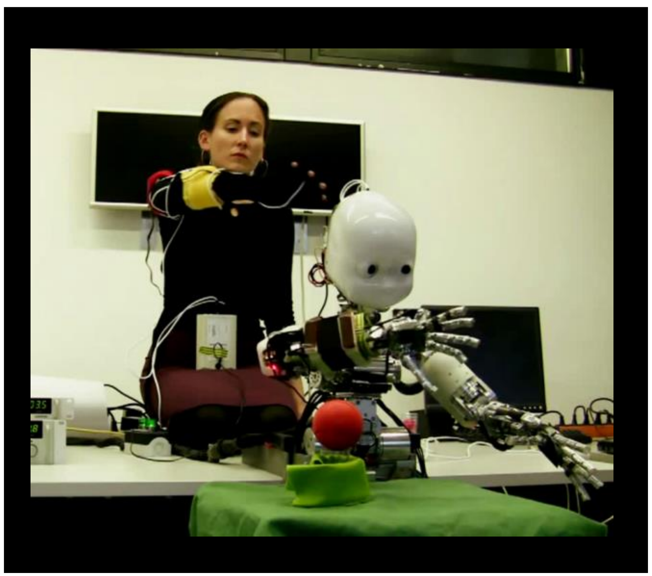
$$\dot{x} = M(\Gamma(x))\, f(x)$$

Guarantees that the robot will never penetrate the obstacle.
Guarantees that the robot will reach the goal.

Learning Dynamical Systems-based Control Laws

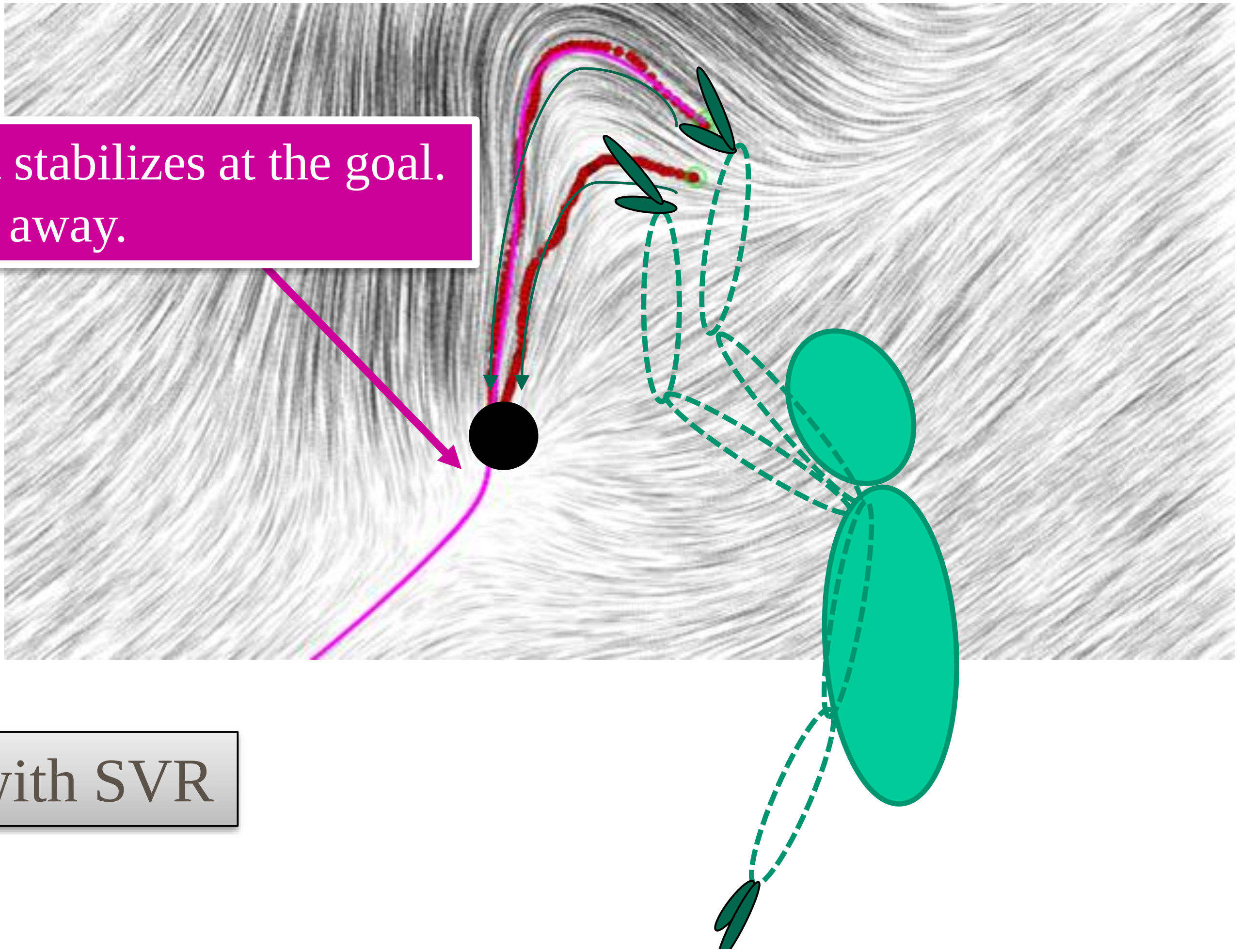
Solution 1:
Provide demonstrations of feasible trajectories





Solution 2:
Generate trajectories from optimal control

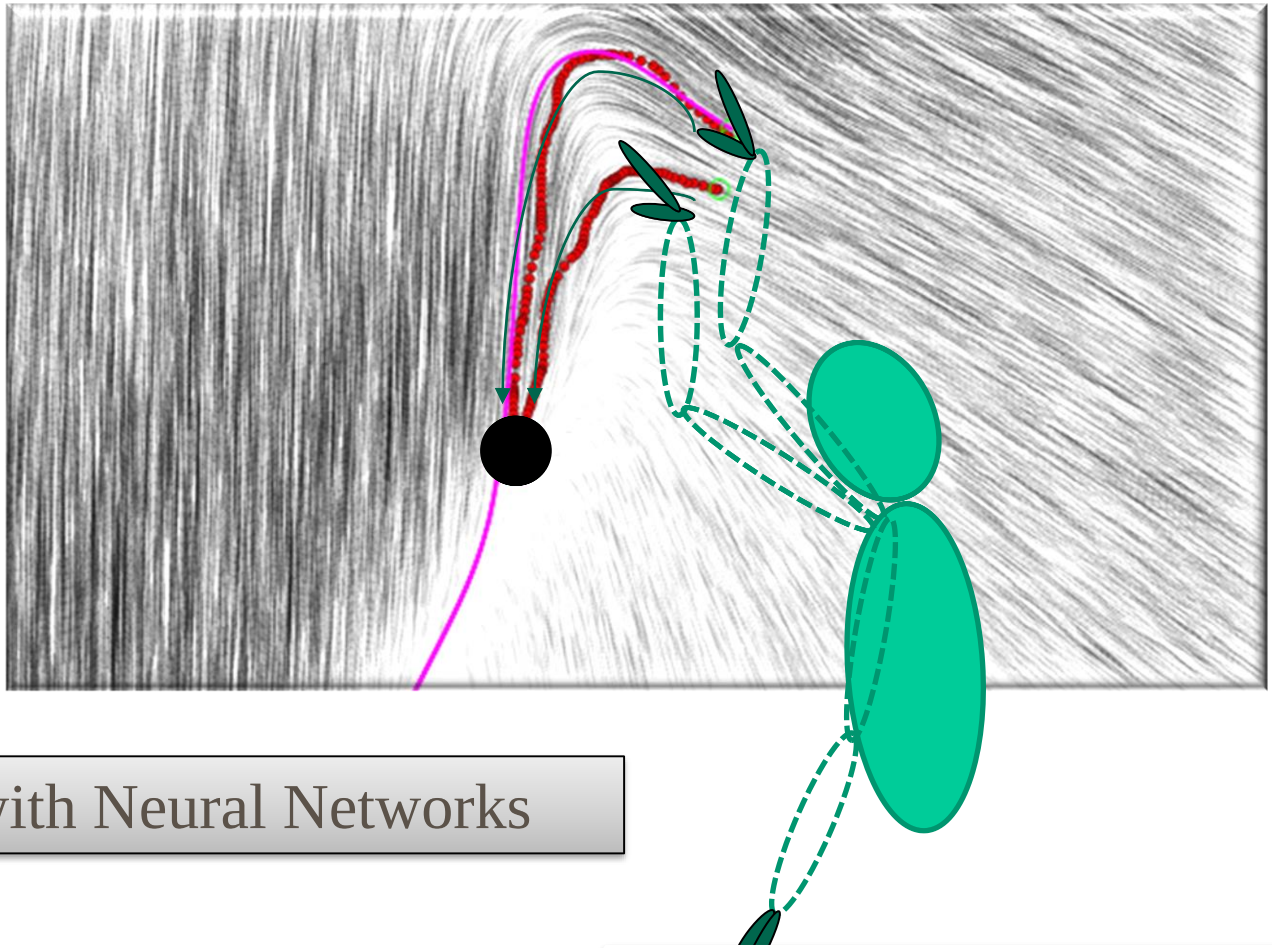
We will see solution 2 in the programming exercises of the course.



Path does not stabilizes at the goal.
Motion drifts away.

Learned with SVR

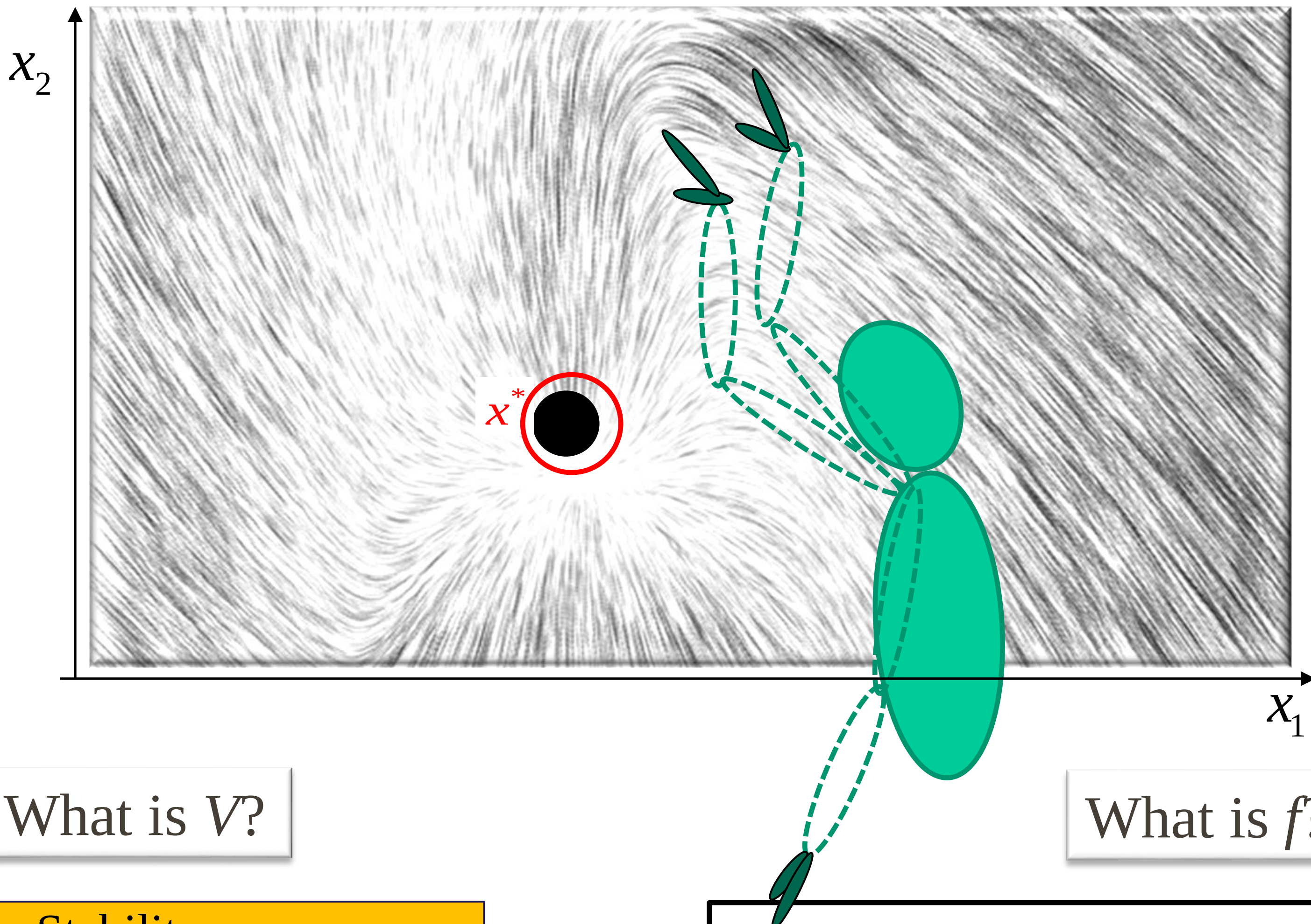
Learn a function: $\dot{x} = f(x)$



Learned with Neural Networks

Learn a function: $\dot{x} = f(x)$

Drifts happens as there is no constraint in the optimization of SVR or NN that forces the learned model to stop at the goal.



What is V?

What is f?

Lyapunov Stability:

$\exists V(x)$ positive,

s.t. $V(x^*)=0$ & $\dot{V}(x)<0 \; \forall x \neq x^*$

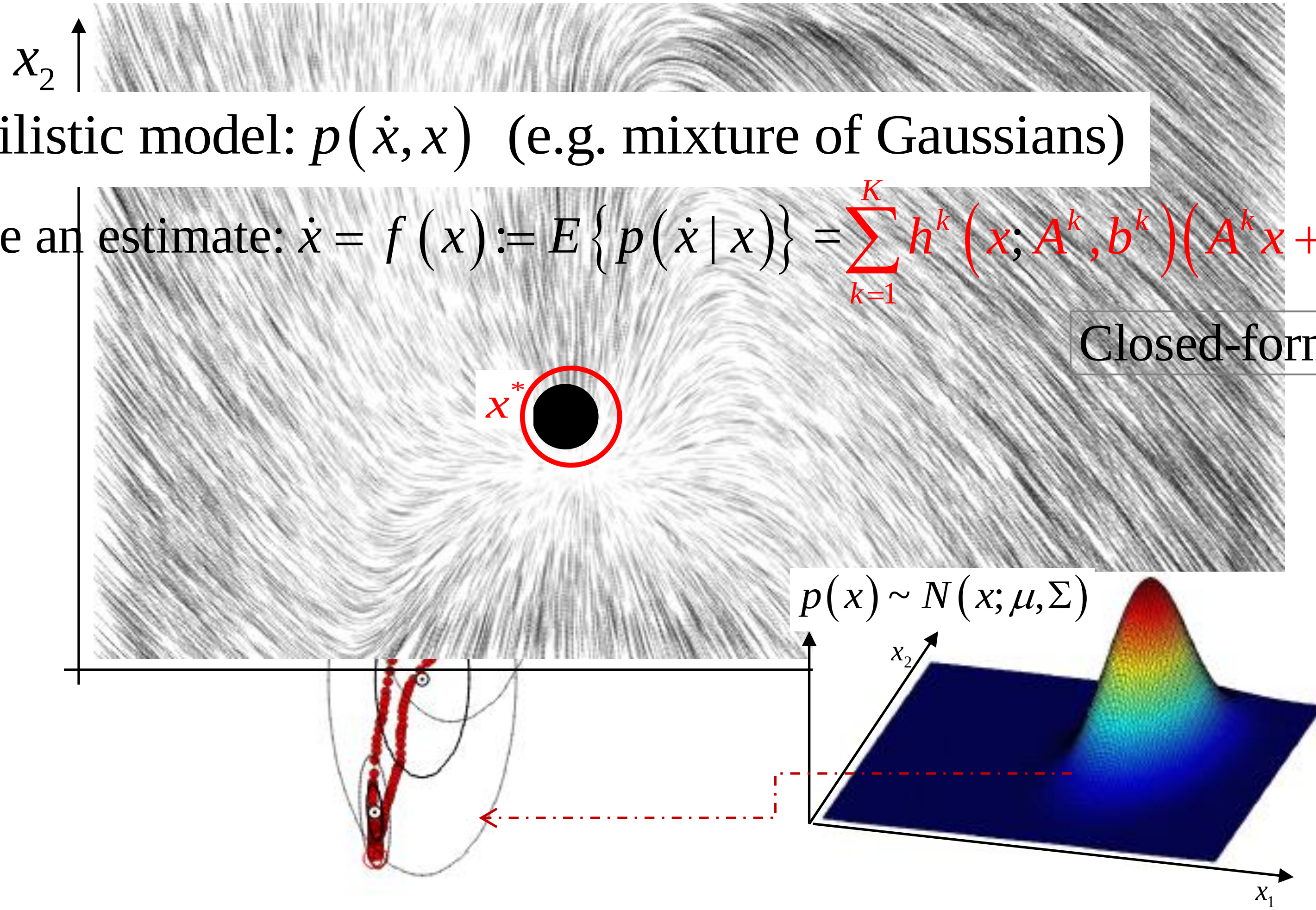
Convergence to a fixed point.

$$\dot{x}^* = f(x^*) = 0, \qquad \lim_{t \to \infty} \dot{x} = 0$$

Probabilistic model: $p(\dot{x}, x)$ (e.g. mixture of Gaussians)

Generate an estimate: $\dot{x} = f(x) := E\{p(\dot{x} \mid x)\} = \sum_{k=1}^{K} h^k(x; A^k, b^k)(A^k x + b^k)$

Closed-form solution



2D projection of a normal distribution

Stable Estimator of Dynamical Systems (SEDS)

Probabilistic model: $p(\dot{x}, x)$ (e.g. mixture of Gaussians)

Generate an estimate: $\dot{x} = f(x) := E\{p(\dot{x} \mid x)\} = \sum_{k=1}^{K} h^k\left(x; A^k, b^k\right)\left(A^k x + b^k\right)$

Closed-form solution

Choose Lyapunov function

$$V(x) = \left(x - x^*\right)^T \left(x - x^*\right)$$

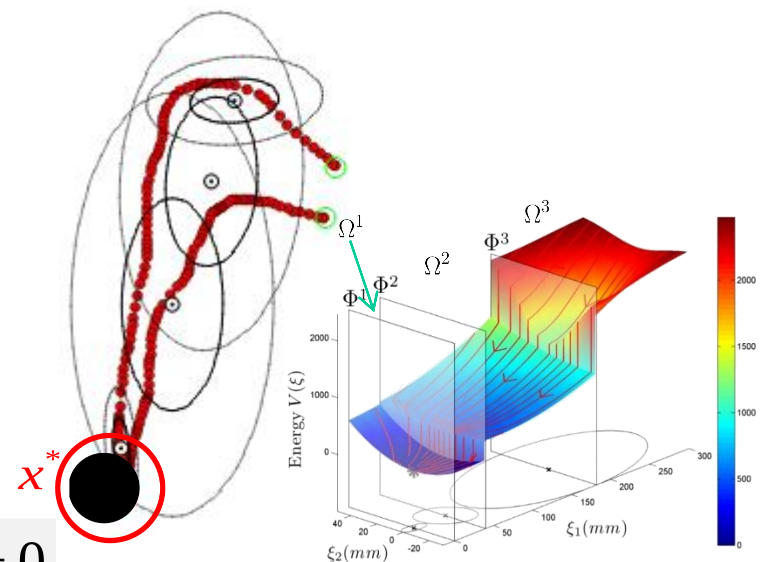
Stability at target:

$(a)\ b^k = -A^k x^*$

Energy decreases

$$A^k + \left(A^k\right)^T \prec 0 \quad \forall k$$

x_2



$f(x) = 0$

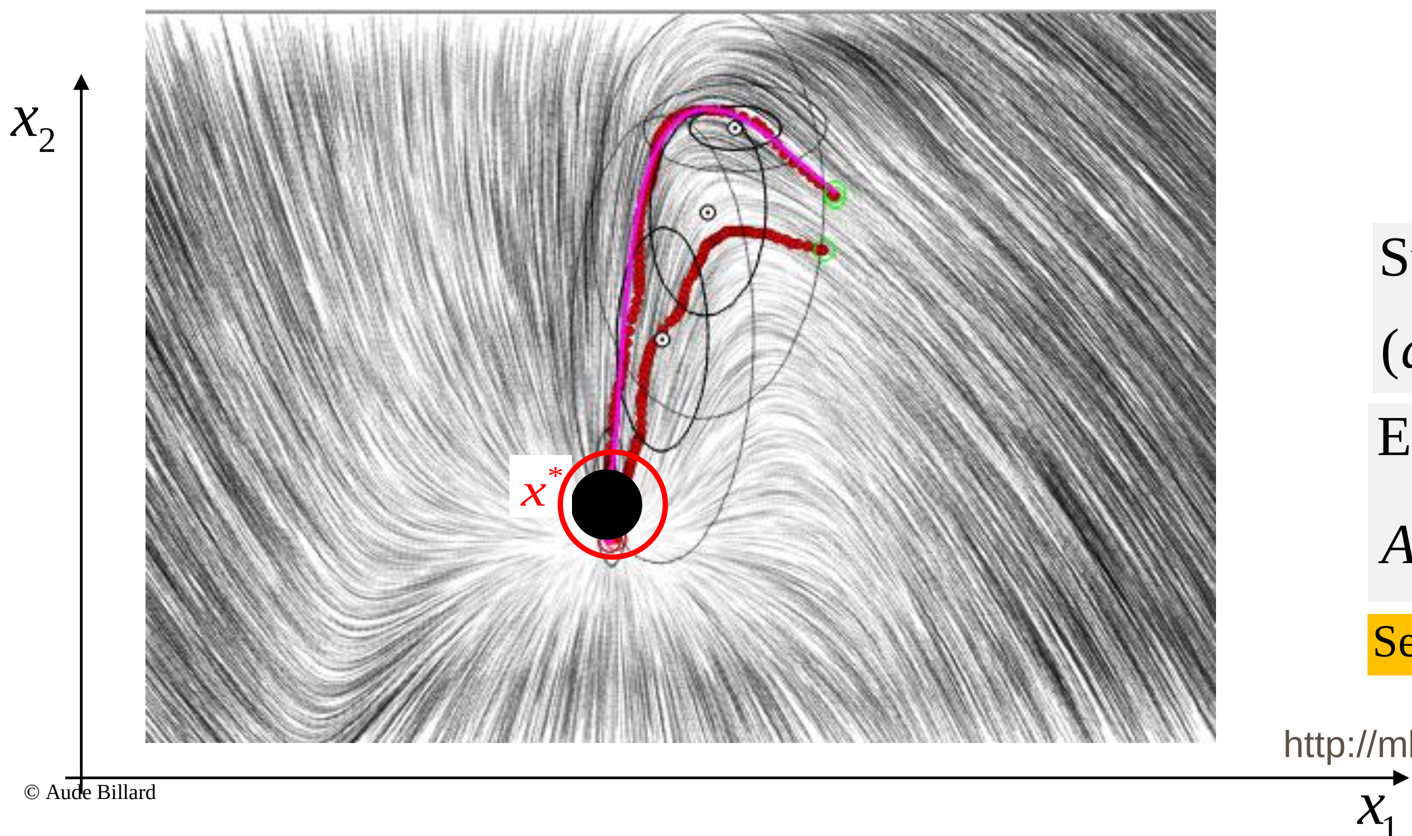
Constrained optimization problem (maximize likelihood under stability constraints)

Stable Estimator of Dynamical Systems (SEDS)

Probabilistic model: $p(\dot{x}, x)$ (e.g. mixture of Gaussians)

Generate an estimate: $\dot{x} = f(x) := E\{p(\dot{x} \mid x)\} = \sum_{k=1}^{K} h^k(x; A^k, b^k)(A^k x + b^k)$

Closed-form solution



Stability at target:

$(a)\ b^k = -A^k x^*$

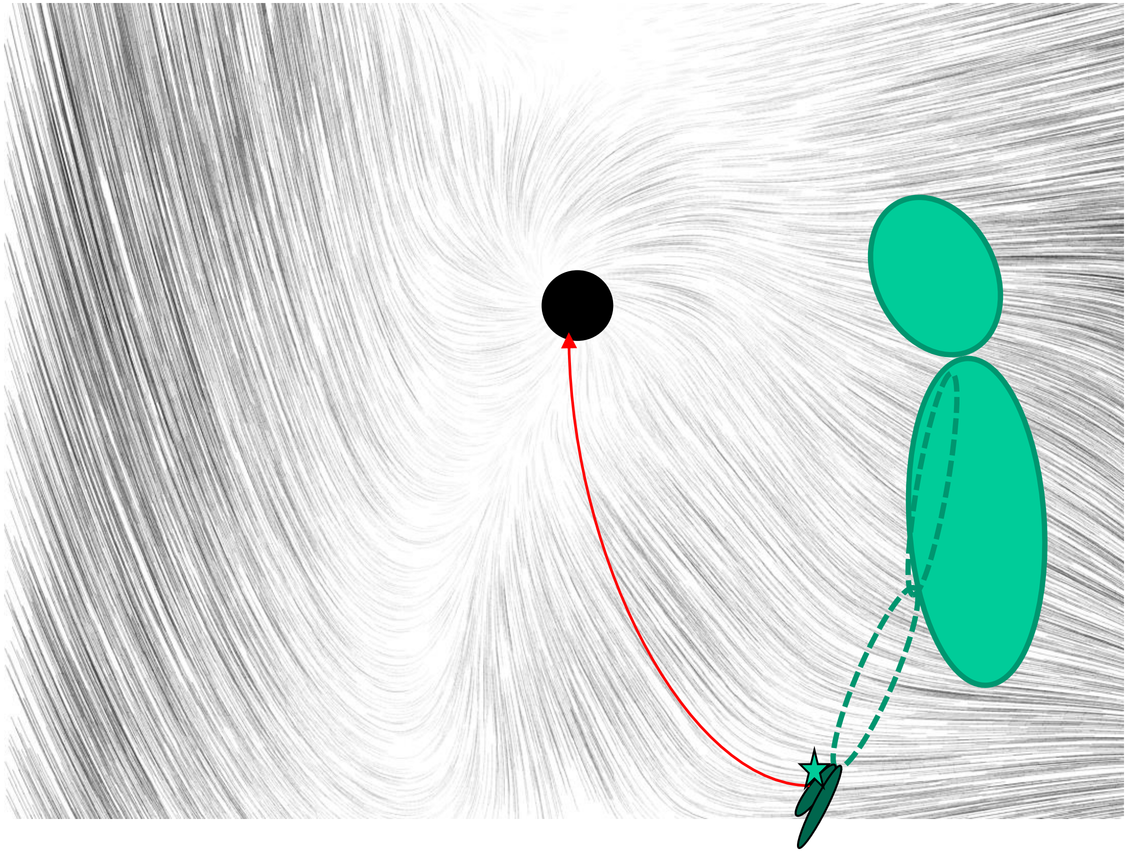
Energy decreases

$A^k + (A^k)^T \prec 0 \quad \forall k$

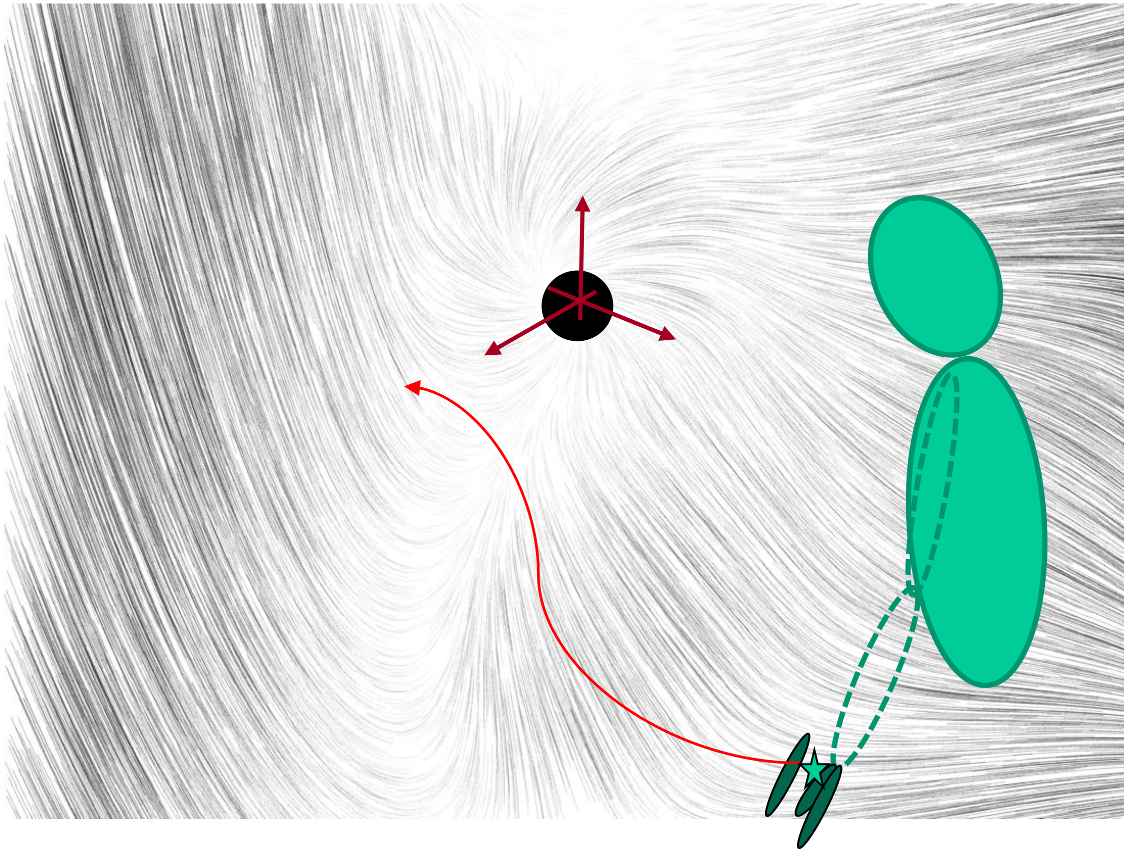
See Chapter 3 of Book

http://mldemos.epfl.ch

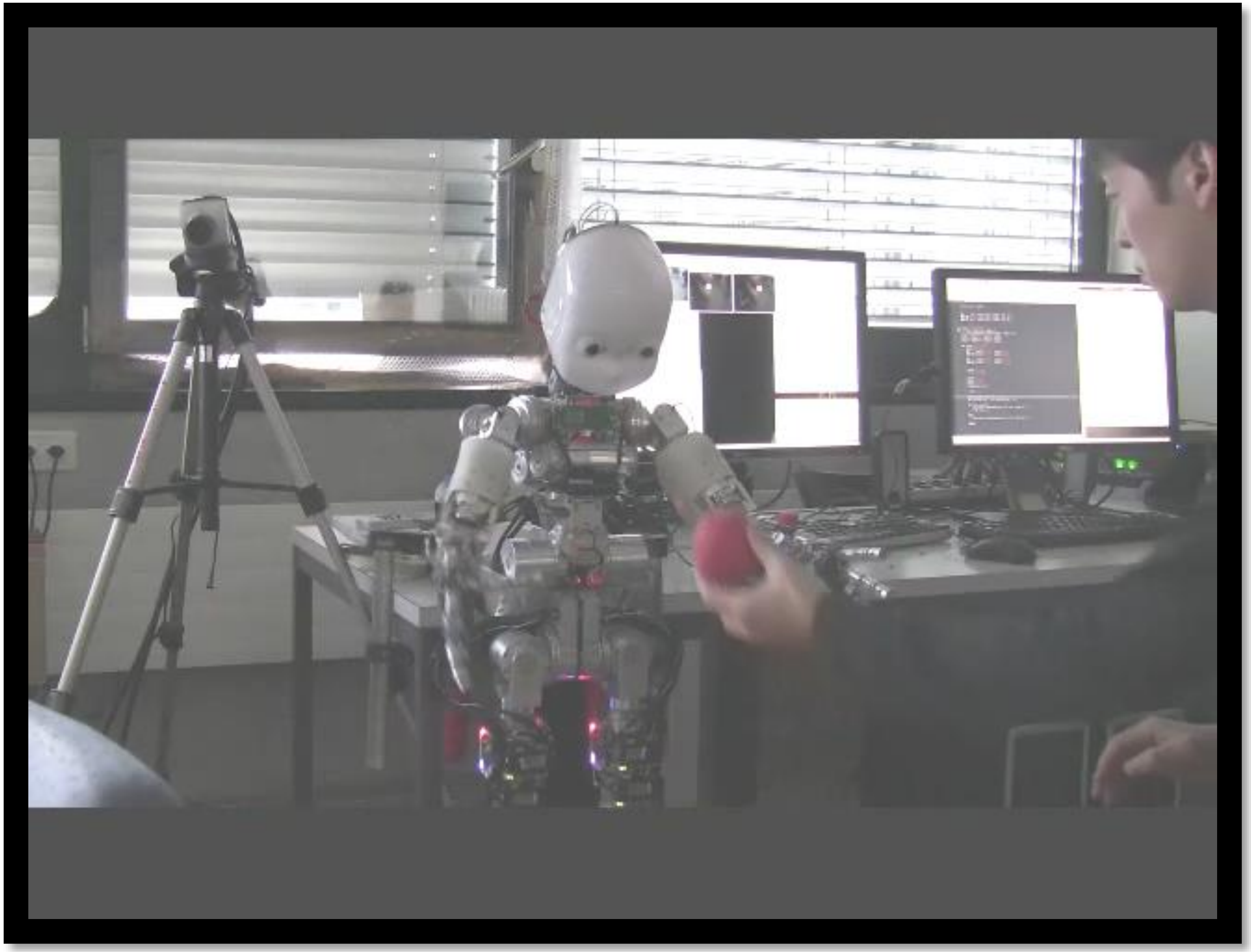
What if the target moves?

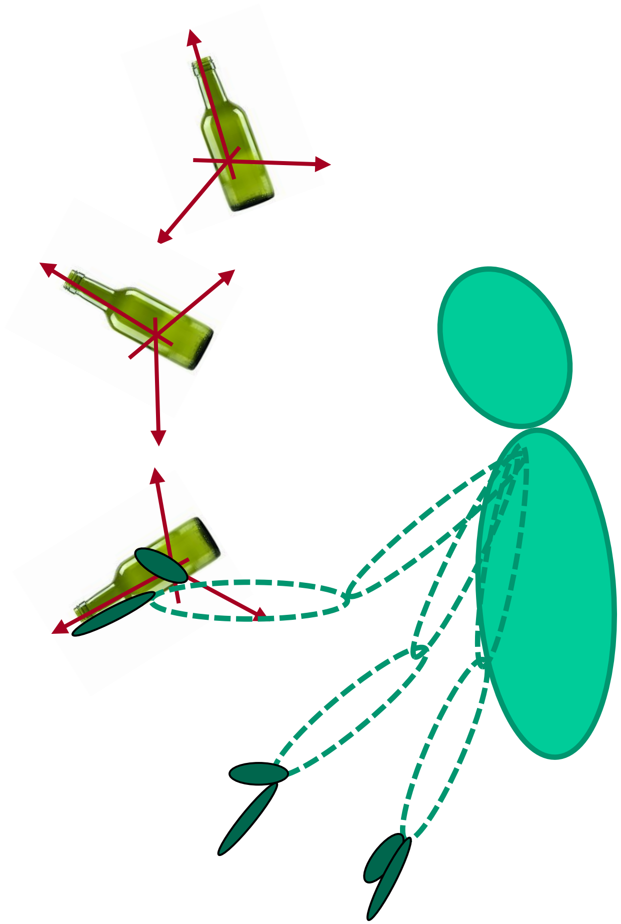


What if the target moves?



Fixed point at the origin: $x^* = O$



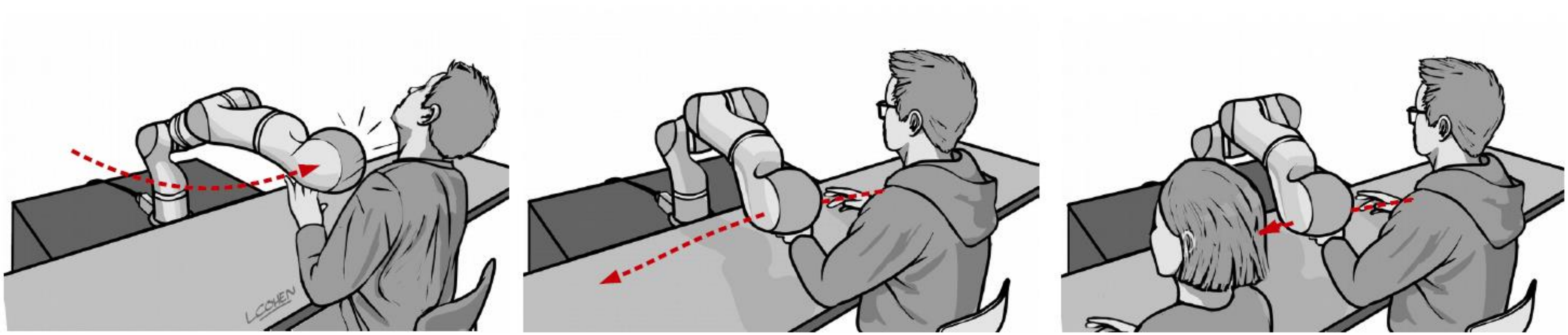


Requires also multi-attractor system, see Chapter 4 of Book

From Path Planning to Real-Time Force Control

Safety – Compliant Control

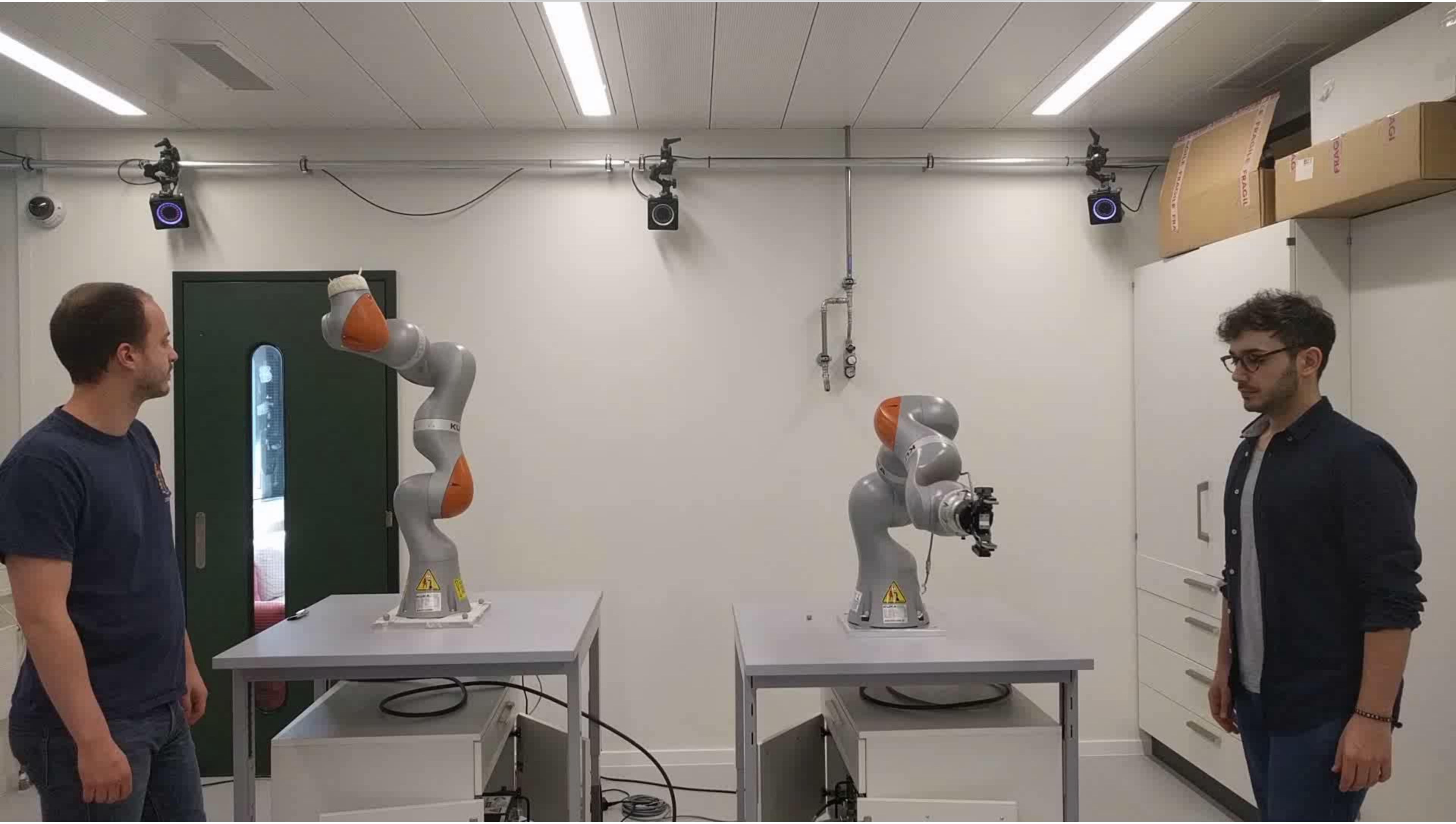
Robots must remain safe to interact with

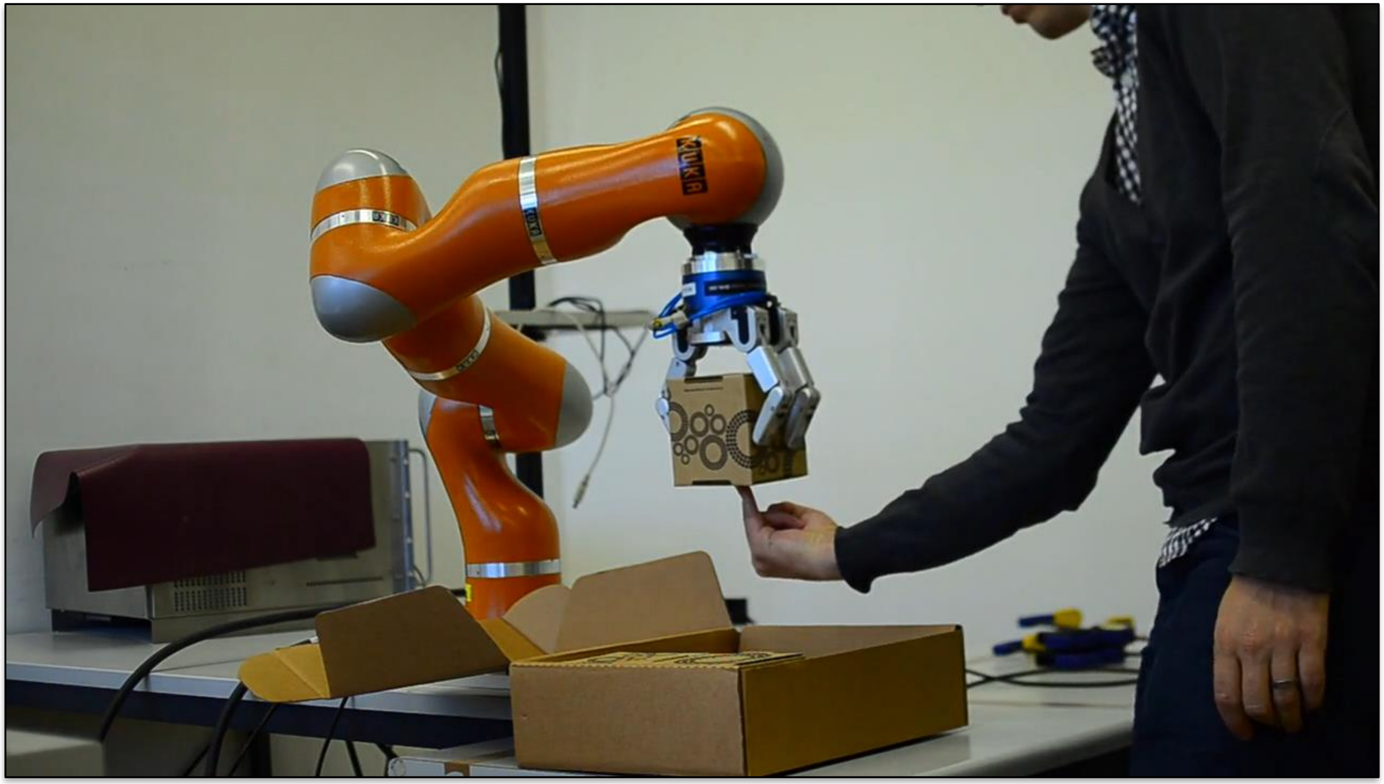


A. Stiff robot: collision **B.** Compliant robot **C.** Compliant robot: collision risk during change of direction

EPFL

Safety – Compliant Control



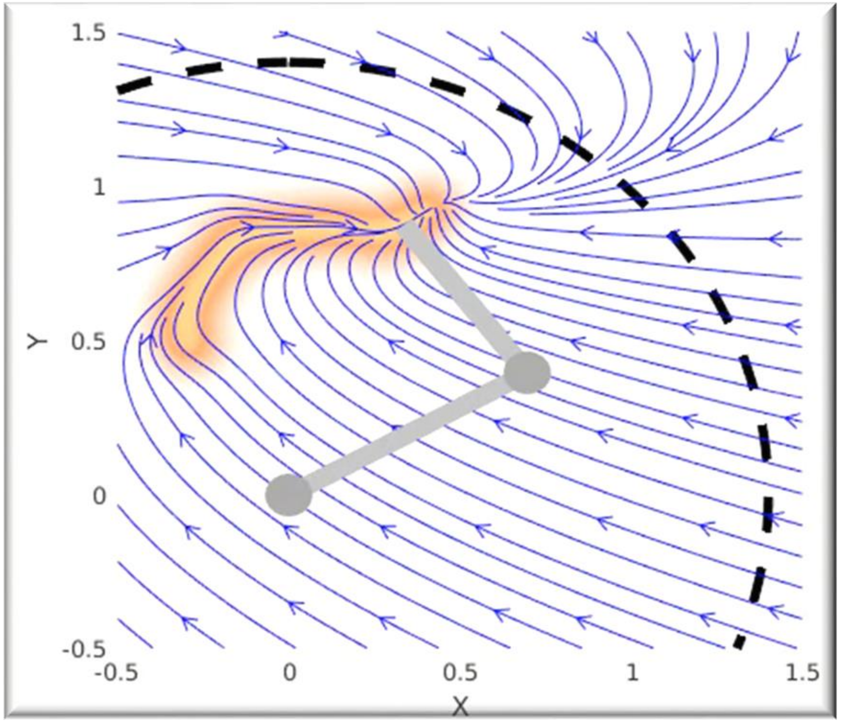


***React** continuously*

and immediately

Impedance control + On-line trajectory planning with dynamical systems

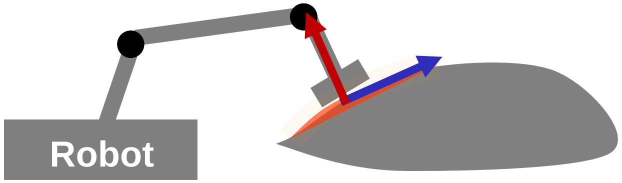
See Chapter 10 of Book



***React** continuously*
and immediately

Impedance control + On-line trajectory planning with dynamical systems

See Chapter 11 of Book



Position controlled direction

Force controlled direction

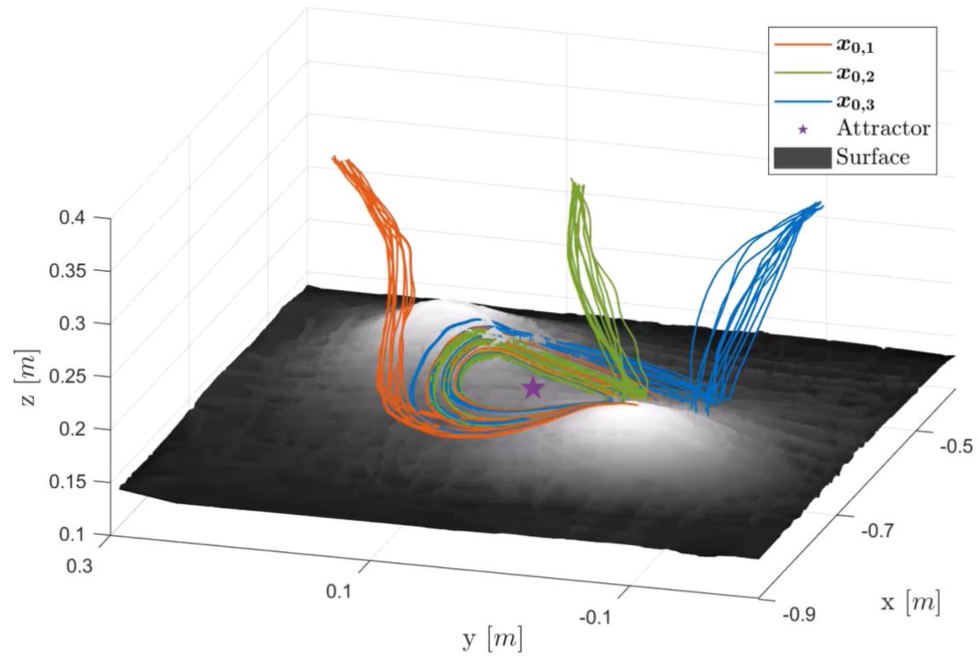
$$D(x)\left(\dot{x}-f(x)\right)=\tau$$

$$f(x)=f_{//}(x)+f_{\mp}(x)$$

Impedance control

On-line trajectory planning with dynamical systems

All the **trajectories recorded** and the **surface model learned** can be found below



$$D(x)(\dot{x} - f(x)) = \tau$$

$$f(x) = f_{//}(x) + f_{\mp}(x)$$

Impedance control

On-line trajectory planning with dynamical systems

Book Sections & Complements Related to this Lecture

Relevant sections of the book *Learning and adaptive control for robots, MIT Press* :

- Chapter 1: Using and Learning Dynamical Systems for Control
- Appendix C 2.3: Inverse Kinematics
- Appendix C.1: Multi-rigid Body Dynamics
- Appendix C.2.2: Motion control with Dynamical Systems

Complements to refresh your memory on basis of robot control can be found in the Handbook of Robotics – Spring-Verlag – available for free on-line

Part A: Robotics Foundation

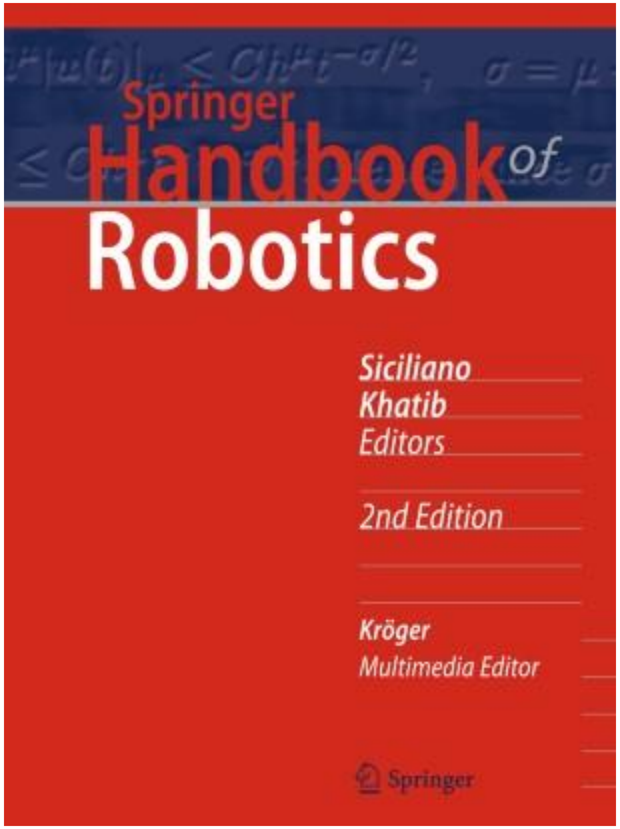
- Section 2 - Kinematics
- Section 7 - Motion Planning

Overview Course & References to Book Sections

WEEK	TOPIC	BOOK Chapter
1	Intro to robot path planning	Chapter 1
2	Acquiring data for learning	Chapter 2
3	Introduction to dynamical systems (DS)	Annexes A
4	Learning control laws with DS	Chapter 3
5	PRACTICE SESSION I	
6	Learning how to modulate a dynamical system	Chapter 8
7	Obstacle avoidance with dynamical systems	Chapter 9
8	PRACTICE SESSION II	
9	Impedance control with dynamical systems	Chapter 10
10	Force control with dynamical systems	Chapter 11
11	PRACTICE SESSION III	
12	Extensions & other application to learning with DS PRACTICE SESSION III CONTINUED	Selections from Ch. 4,5,6&7* * Not exam material
13	Overview and Exam Preparation PRACTICE SESSION III CONTINUED	

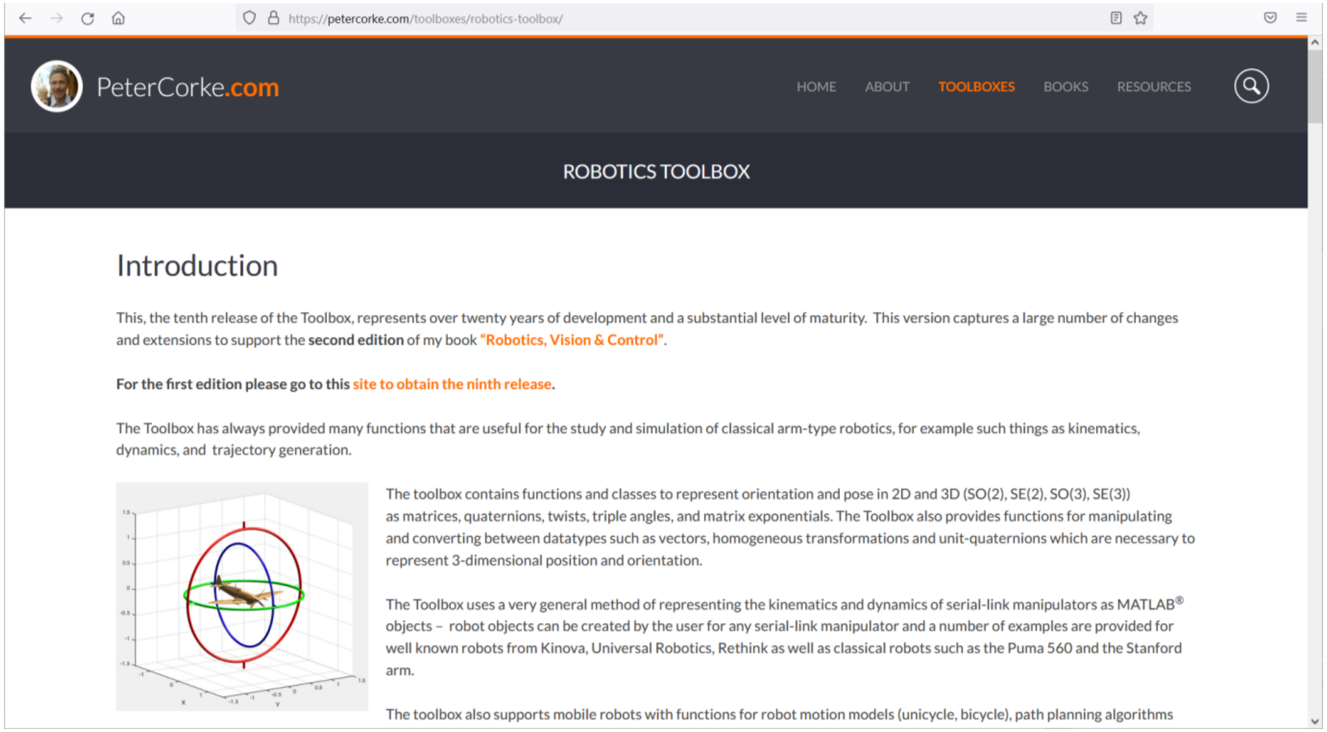
Good Robotics Resources

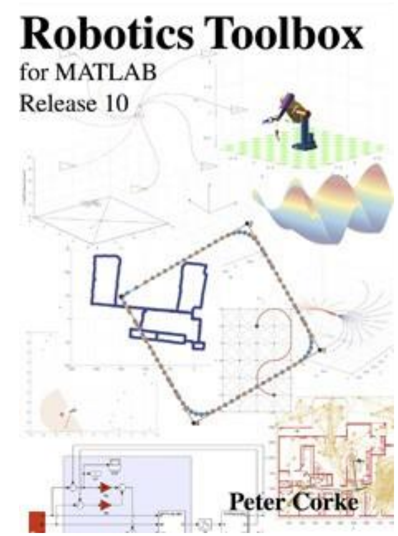
Free PDF https://link.springer.com/content/pdf/10.1007%2F978-3-319-32552-1.pdf

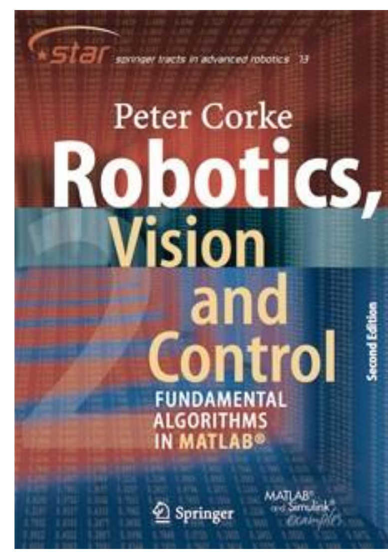


Comprehensive overview of fundamental algorithms in Robotics and of recent advances in robot learning, human-robot interaction, design of soft robots, etc.

Good Robotics Software Resources







Large set of libraries to control robots, large set of robots

https://petercorke.com/toolboxes/robotics-toolbox/