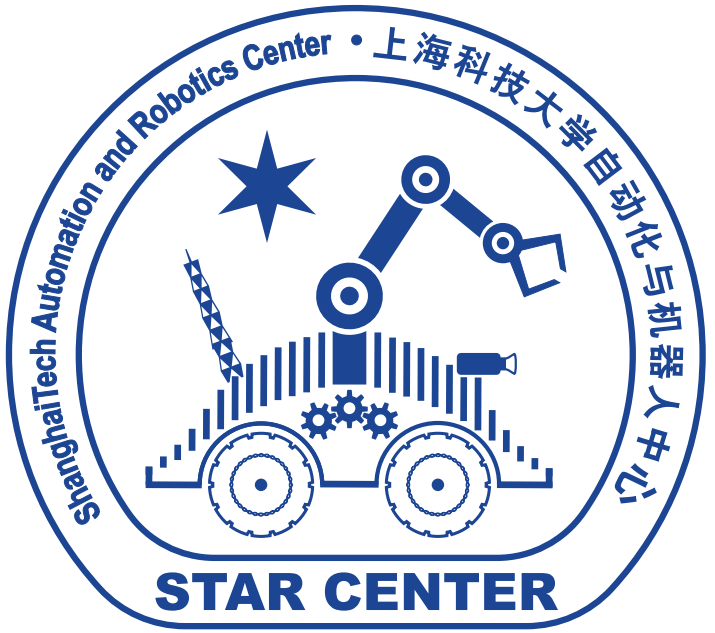
上海科技大学

ShanghaiTech University

CS283: **Robotics Fall 2020: Summary**

Sören Schwertfeger / 师泽仁

ShanghaiTech University



Admin

- HW 4 is (finally) due Dec 20
- HW 5 is published
  - Due Jan 08
    - You are encouraged to schedule your slot ealier!
  - Fixed time of 2 to max 3 hours
  - Work in your project group
  - Play with the real Fetch robot
  - Basically: beloved HW 3 with a real robot
- Autonomous Driving lecture this Thursday!

Project conclusion

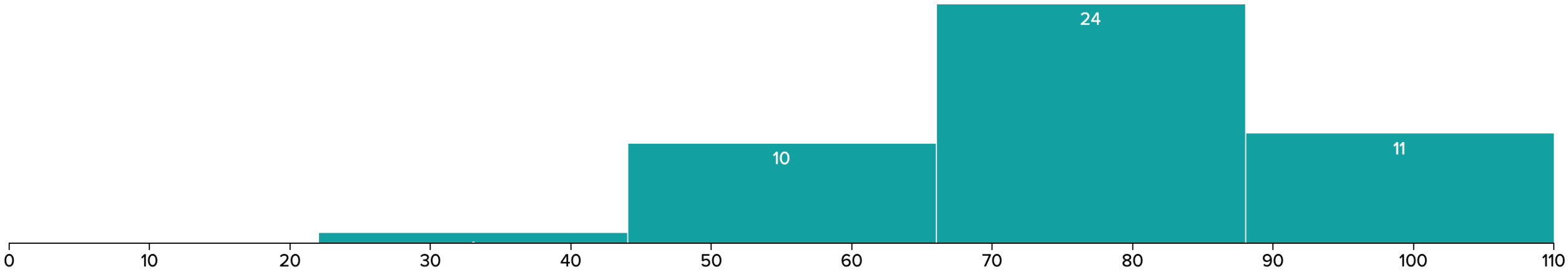
- Due Jan 8:
  - Project Report:
    - Like intermediate report
    - With nice results and proper quantitative evaluation
    - Make look like a scientific paper
    - Use bibtex!
    - Put into git (folder: doc/final)
      - Include everything that is needed to generate the report in the git!
        - So don’t forget images/ the bib file
  - Project Demo:
    - Make an appointment with Prof. Schwertfeger and your TA to show the final demo of your project

Project Webpage

- Write a text about your project for the general public – not too technical – not too many details
  - Some details can be written
- Provide a few images with captions
- Put into your group git (folder: doc/webpage)
- Prof. Schwertfeger will upload the data to the website – e.g. look at : https://robotics.shanghaitech.edu.cn/teaching/robotics2017
- Also make a NICE video about your project. 4-8 minutes. Leave the video at good quality – size maybe 100 – 300 MB (MP4) – Prof. Schwertfeger will compress it to make a web version
  - Avoid showing other people; do not talk in the video; do not add music; add a title page
- Upload with your project name to: https://star-center.shanghaitech.edu.cn/seafile/u/d/7a8f71c6f6274a5dab41/

Midterm

- Regrade requests are open till this Friday.
- Solution pdf is available on piazza.



MINIMUM	MEDIAN	MAXIMUM	MEAN	STD DEV
37.0	74.75	101.0	74.6	16.1

Final

- Most likely in 2nd Final week (Jan 4 to 8)

- Content:
  - All lectures
    - Take a look at facts, algorithms, concepts
  - Take a look at the homeworks again
  - Sample exam: https://robotics.shanghaitech.edu.cn/sites/default/files//files/final_Example.pdf

- You are allowed to bring **3** A4 sheets (so 6 pages) of info to the exams. You can write/ print anything on those sheets. On top of **every page** (so 6 times) there needs to be your **name (pinyin), student ID and ShanghaiTech email** address. We will check every cheat sheet before the exam and **confiscate** every sheet without name or with a name that is not yours.

- No electronics/ calculator/ smartwatch allowed

Midterm - Final

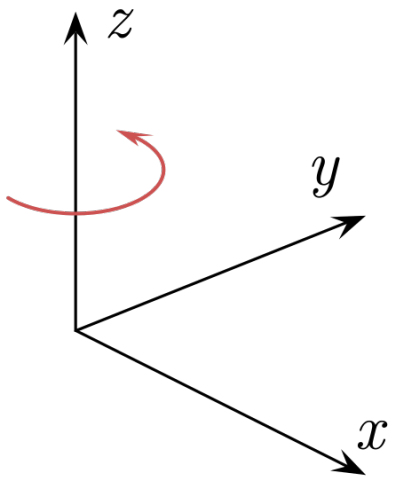
- Questions in final can be:
  - Similar to the midterm
  - New questions (e.g. other algorithm)

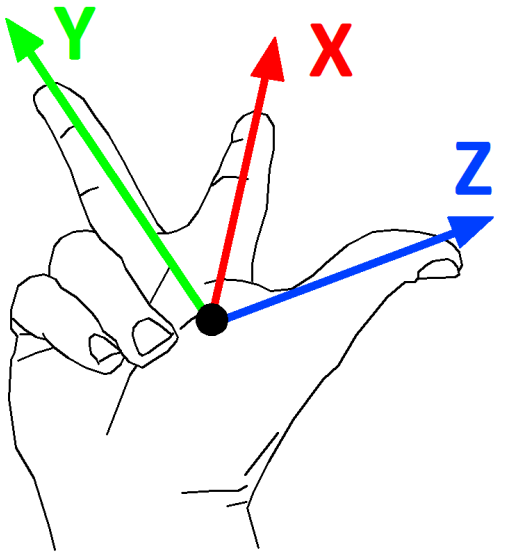
- => learn for your Midterm mistakes!

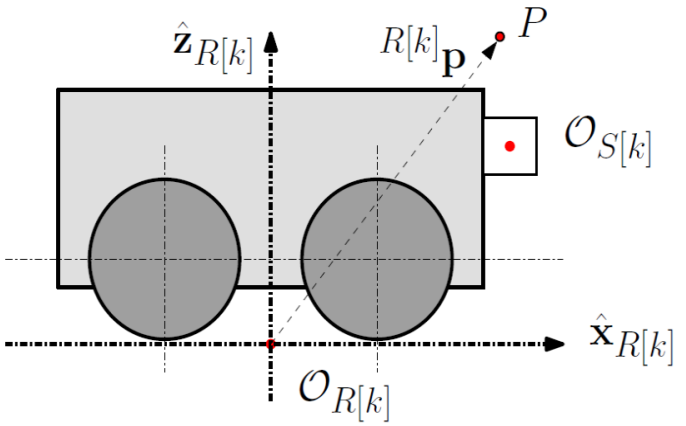
- We may include questions from the guest lectures!

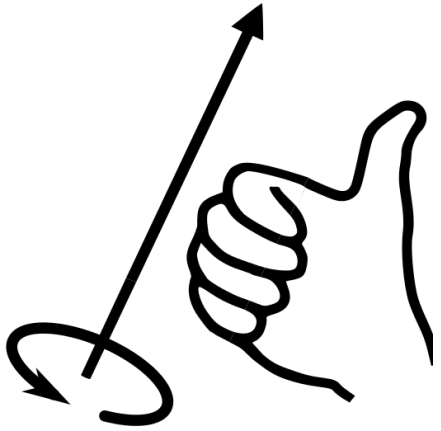
Right Hand Coordinate System

- Standard in Robotics
- Positive rotation around X is anti-clockwise
- Right-hand rule mnemonic:
  - Thumb: z-axis
  - Index finger: x-axis
  - Second finger: y-axis
  - Rotation: Thumb = rotation axis, positive rotation in finger direction
- Robot Coordinate System:
  - X front
  - Z up (Underwater: Z down)
  - Y ???





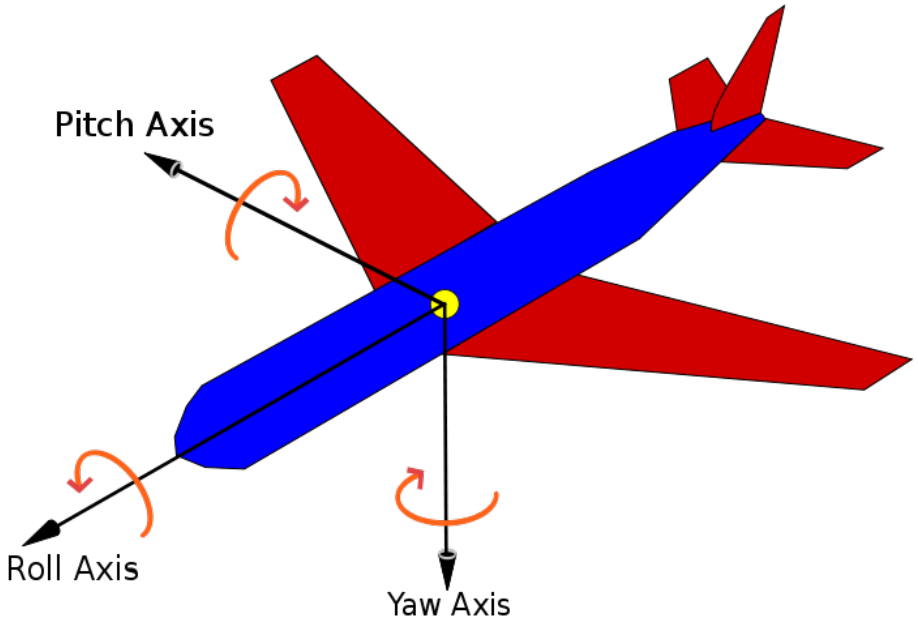




Right Hand Rule http://en.wikipedia.org/wiki/Right-hand_rule

3D Rotation

- Euler angles: Roll, Pitch, Yaw
  - ☹ Singularities
- Quaternions:
  - Concatenating rotations is computationally faster and numerically more stable
  - Extracting the angle and axis of rotation is simpler
  - Interpolation is more straightforward
  - Unit Quaternion: norm = 1
  - Scalar (real) part: q_0 , sometimes q_w
  - Vector (imaginary) part: q
  - Over determined: 4 variables for 3 DoF

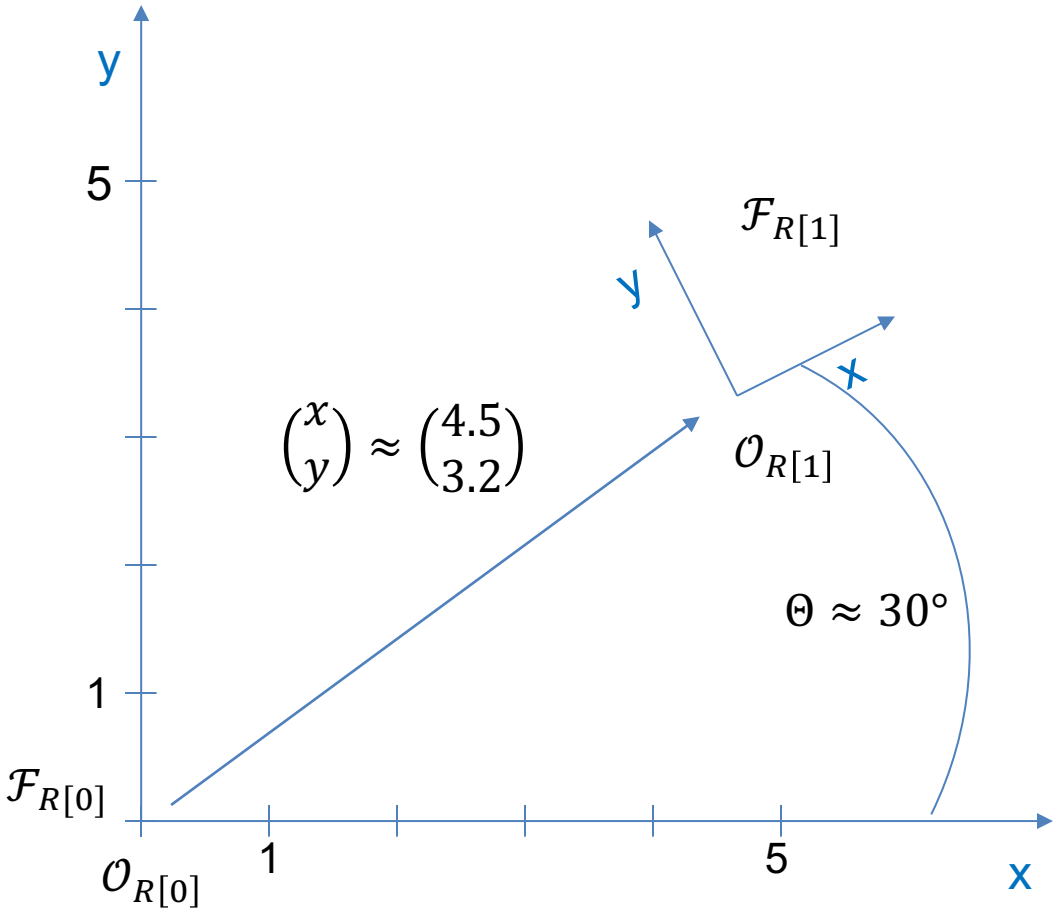


$$\check{\mathbf{p}} \equiv p_0 + p_x\mathbf{i} + p_y\mathbf{j} + p_z\mathbf{k}$$

$$\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -\mathbf{1}$$

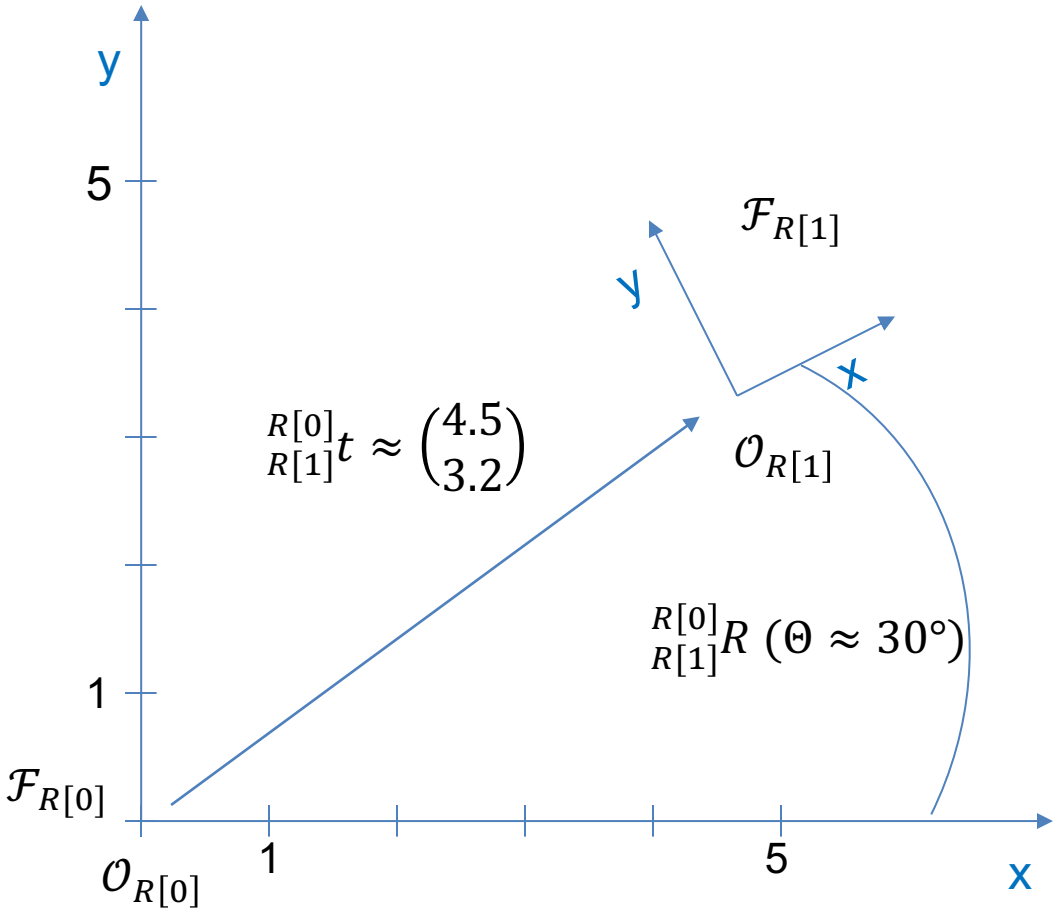
$$\check{\mathbf{q}} = \begin{pmatrix} q_0 & q_x & q_y & q_z \end{pmatrix}^\mathsf{T} \equiv \begin{pmatrix} q_0 \\ \mathbf{q} \end{pmatrix}$$

Position, Orientation & Pose



- **Position**:
  - $\begin{pmatrix} x \\ y \end{pmatrix}$ coordinates of any object or point (or another frame)
  - with respect to (wrt.) a specified frame
- **Orientation**:
  - (Θ) angle of any oriented object (or another frame)
  - with respect to (wrt.) a specified frame
- **Pose**:
  - $\begin{pmatrix} x \\ y \\ \Theta \end{pmatrix}$ position and orientation of any oriented object
  - with respect to (wrt.) a specified frame

Translation, Rotation & Transform



- **Translation**:
  - $\begin{pmatrix} x \\ y \end{pmatrix}$ difference, change, motion from one reference frame to another reference frame
- **Rotation**:
  - (Θ) difference in angle, rotation between one reference frame and another reference frame
- **Transform**:
  - $\begin{pmatrix} x \\ y \\ \Theta \end{pmatrix}$ difference, motion between one reference frame and another reference frame

Transform in 3D

Matrix Euler Quaternion

$$ {}^{G}_{A}\mathbf{T} = \begin{bmatrix} {}^{G}_{A}\mathrm{R} & {}^{G}_{A}\mathrm{t} \\ 0_{1x3} & 1 \end{bmatrix} = \begin{pmatrix} {}^{G}_{A}\mathrm{t} \\ {}^{G}_{A}\Theta \end{pmatrix} = \begin{pmatrix} {}^{G}_{A}\mathrm{t} \\ {}^{G}_{A}\check{\mathrm{q}} \end{pmatrix} $$

$$ {}^{G}_{A}\Theta \triangleq (\theta_r, \theta_p, \theta_y)^T $$

In ROS: Quaternions! (w, x, y, z)
Uses Bullet library for Transforms

Rotation Matrix 3x3

$$ R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} $$

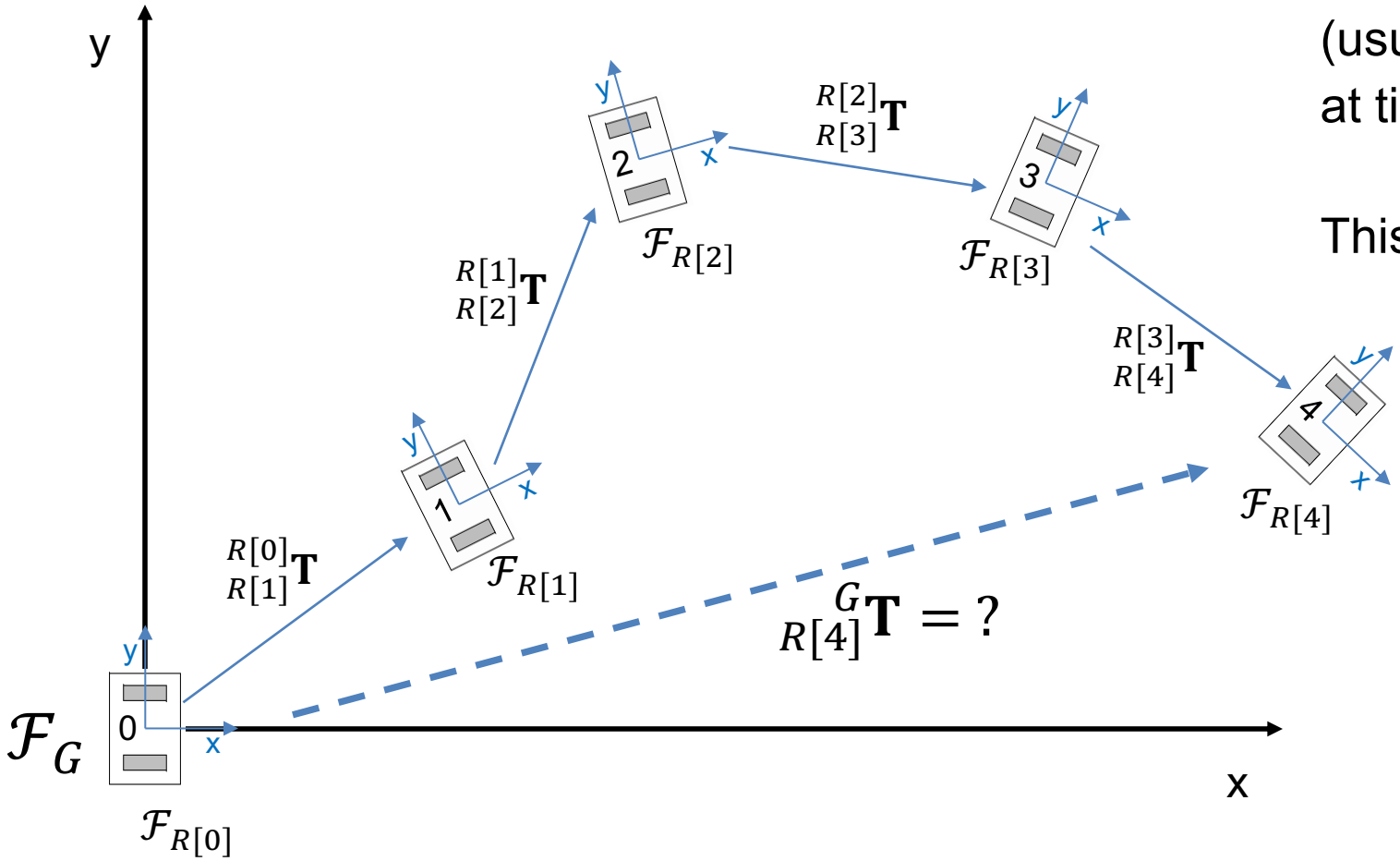
$$ R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} $$

$$ R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} $$

$$ R = R_z(\alpha)\, R_y(\beta)\, R_x(\gamma) $$

yaw = α, pitch = β, roll = γ

Transforms



Where is the Robot now?

The pose of $\mathcal{F}_{R[X]}$ with respect to $\mathcal{F}_G$ (usually = $\mathcal{F}_{R[0]}$) is the pose of the robot at time X.

This is equivalent to $^{G}_{R[X]}\mathbf{T}$

Chaining of Transforms

$$^{G}_{R[X+1]}\mathbf{T} = {}^{G}_{R[X]}\mathbf{T}\ {}^{R[X]}_{R[X+1]}\mathbf{T}$$

often: $\mathcal{F}_G \equiv \mathcal{F}_{R[0]} \Rightarrow {}^{G}_{R[0]}\mathbf{T} = id$

In ROS

- First Message at time 97 : G (frame_id)
- **This** Message at time 103 : X
- Next Message at time 107 : X+1

$$ {}^{G}_{R[X+1]}\mathbf{T} = {}^{G}_{R[X]}\mathbf{T}\ {}^{R[X]}_{R[X+1]}\mathbf{T} $$

$$ {}^{R[X]}_{R[X+1]}t_x $$

$$ {}^{R[X]}_{R[X+1]}t_y $$

$$ {}^{R[X]}_{R[X+1]}\Theta $$

```
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
geometry_msgs/Pose2D pose2D
  float64 x
  float64 y
  float64 theta
```

- Take a look at the other related Pose or Transform messages in ROS!

Questions many students got wrong

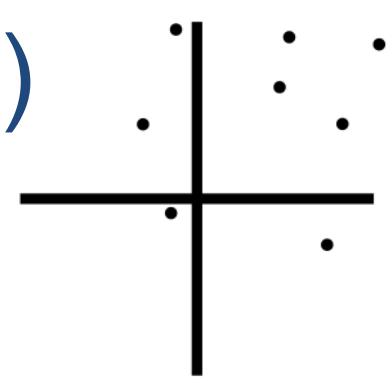
- # of DoF of 3D transform?
  - 6
- ROS tf rotation:
  - Quaternion: x, y, z, w
- TF:
  - frame_id: the name of the parent frame
    child_frame_id: this frame
    (stamp: the time)
- Chaining (remember the name!)

Classification of Sensors

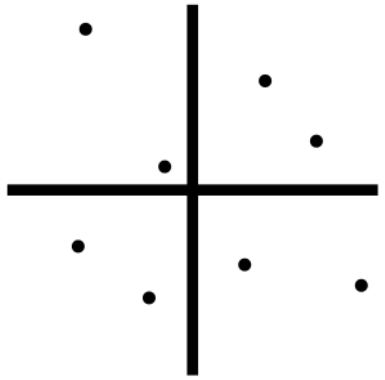
- What:
  - Proprioceptive sensors
    - measure values internally to the system (robot),
    - e.g. motor speed, wheel load, heading of the robot, battery status
  - Exteroceptive sensors
    - information from the robots environment
    - distances to objects, intensity of the ambient light, unique features.
- How:
  - Passive sensors
    - Measure energy coming from the environment
  - Active sensors
    - emit their proper energy and measure the reaction
    - better performance, but some influence on environment

In Situ Sensor Performance (2)

- Error / Accuracy
  - How close to true value
- Precision
  - Reproducibility

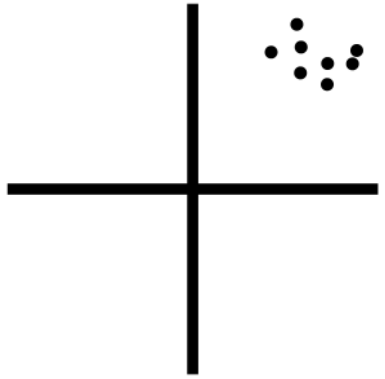


(a) Low precision and low accuracy

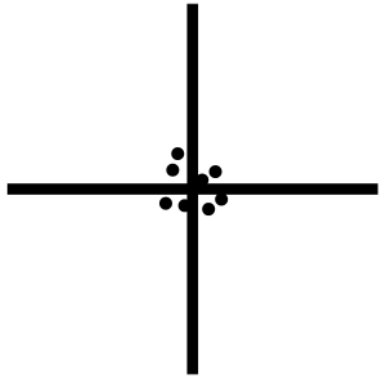


(b) Low precision and high accuracy

- Systematic error -> deterministic errors
  - caused by factors that can (in theory) be modeled -> prediction
  - e.g. calibration of a laser sensor or of the distortion cause by the optic of a camera



(c) High precision and low accuracy

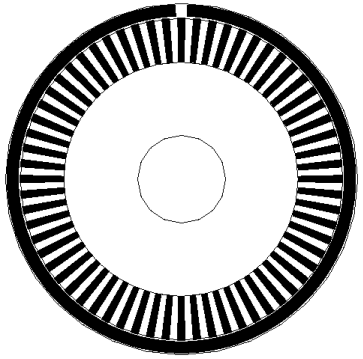


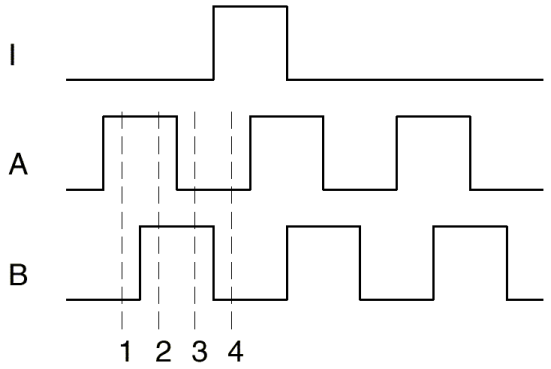
(d) High precision and high accuracy

- Random error -> non-deterministic
  - no prediction possible
  - however, they can be described probabilistically
  - e.g. Hue instability of camera, black level noise of camera ..

Wheel / Motor Encoders

- measure position or speed of the wheels or steering
- integrate wheel movements to get an estimate of the position -> odometry
- optical encoders are proprioceptive sensors
- typical resolutions: 64 - 2048 increments per revolution.
  - for high resolution: interpolation
- optical encoders
  - regular: counts the number of transitions but cannot tell the direction of motion
  - quadrature: uses two sensors in quadrature-phase shift. The ordering of which wave produces a rising edge first tells the direction of motion. Additionally, resolution is 4 times bigger
  - a single slot in the outer track generates a reference pulse per revolution

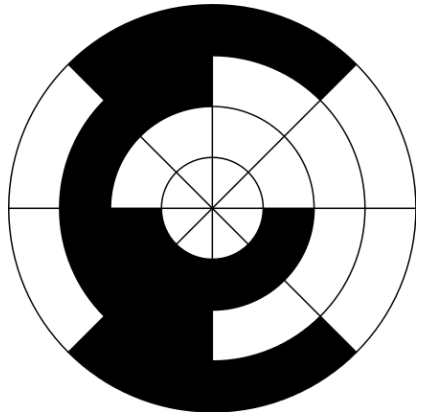




State	Ch A	Ch B
S_1	High	Low
S_2	High	High
S_3	Low	High
S_4	Low	Low

Gray Encoder

http://en.wikipedia.org/wiki/Gray_code

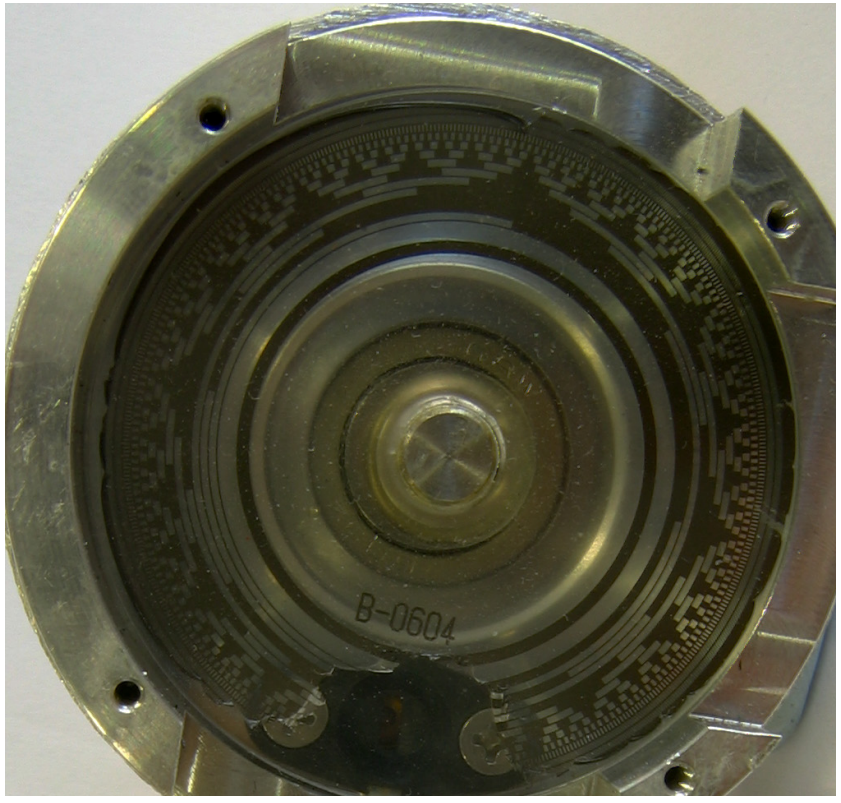


- Aka: reflected binary code, Gray Code
  - Binary numeral system where two successive values differ in only one bit
  - Also used for error correction in digital communications

Only 16% got it right…

- **Absolute position encoder**
  - Normal binary => change from 011 to 100
  - 2 bits change – NEVER simultaneously =>
  - 011 -> 111 -> 101 -> 100 or
  - 011 -> 010 -> 110 -> 100 ….
  - => wrong encoder positions might be read
  - Gray encoding: only one bit change!

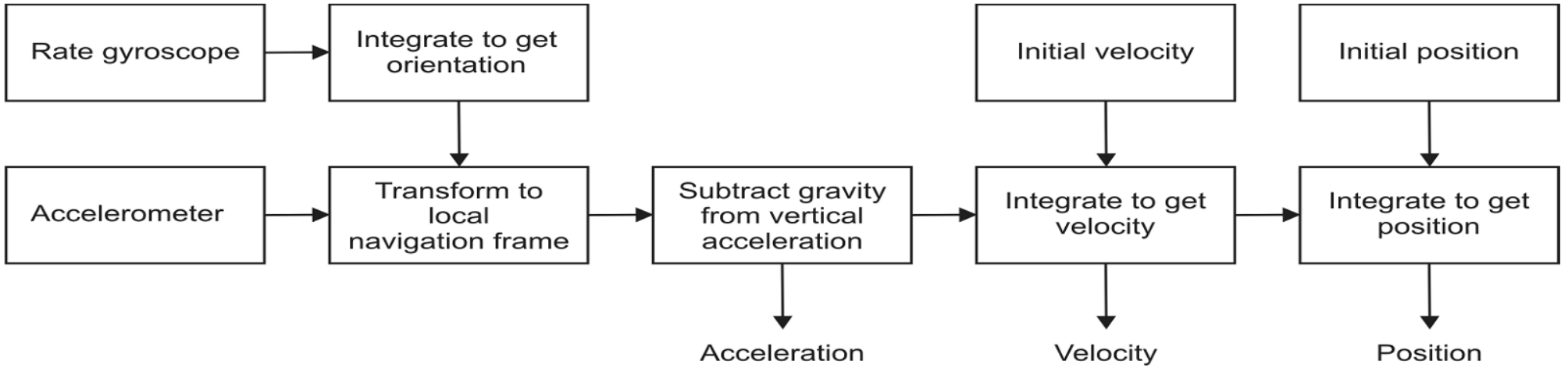
Dec	Gray	Binary
0	000	000
1	001	001
2	011	010
3	010	011
4	110	100
5	111	101
6	101	110
7	100	111



Inertial Measurement Unit (IMU)

- Device combining different measurement systems:
  - Gyroscopes, Accelerometers, Compass
- Estimate relative position (x, y, z), orientation (roll, pitch, yaw), velocity, and acceleration
- Gravity vector is subtracted to estimate motion
  - Initial velocity has to be known

Xsens MTI



IMU Error and Drift

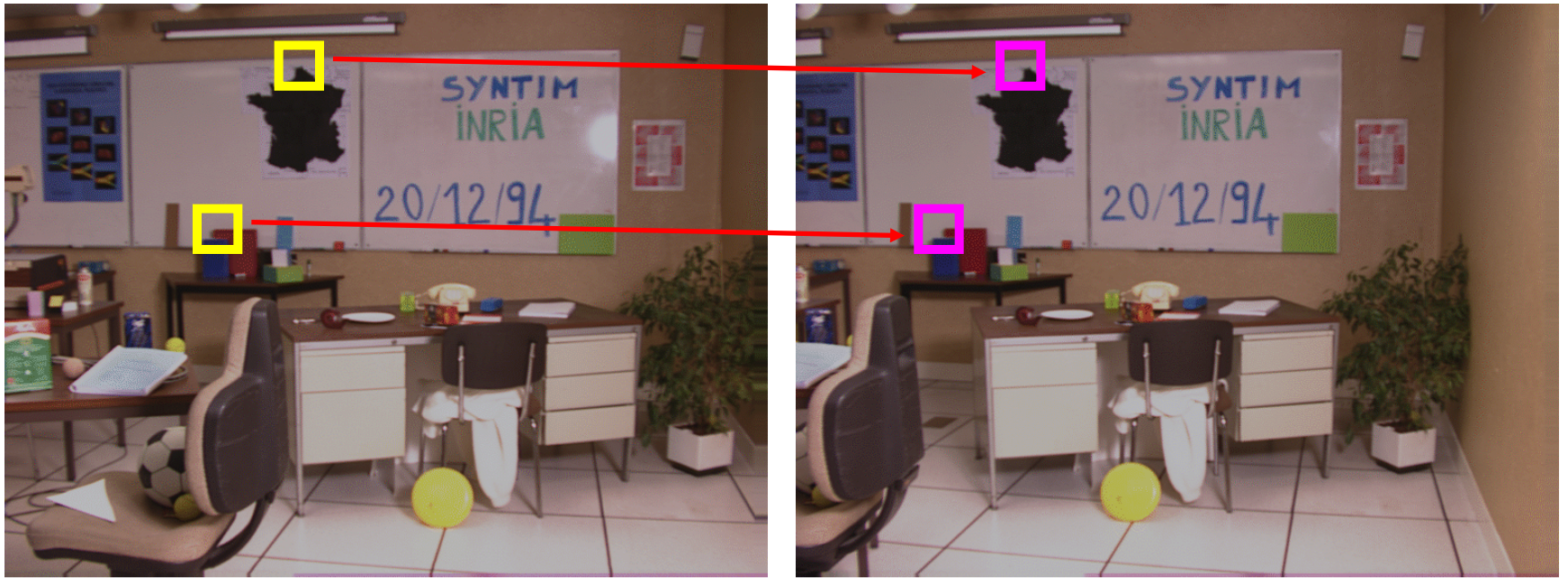
- Extremely sensitive to measurement errors in gyroscopes and accelerometers:
  - drift in the gyroscope unavoidably =>
  - error in orientation relative to gravity =>
  - incorrect cancellation of the gravity vector.

- Accelerometer data is integrated twice to obtain the position => gravity vector error leads to quadratic error in position.

- All IMUs drift after some time
  - Use of external reference for correction:
  - compass, GPS, cameras, localization

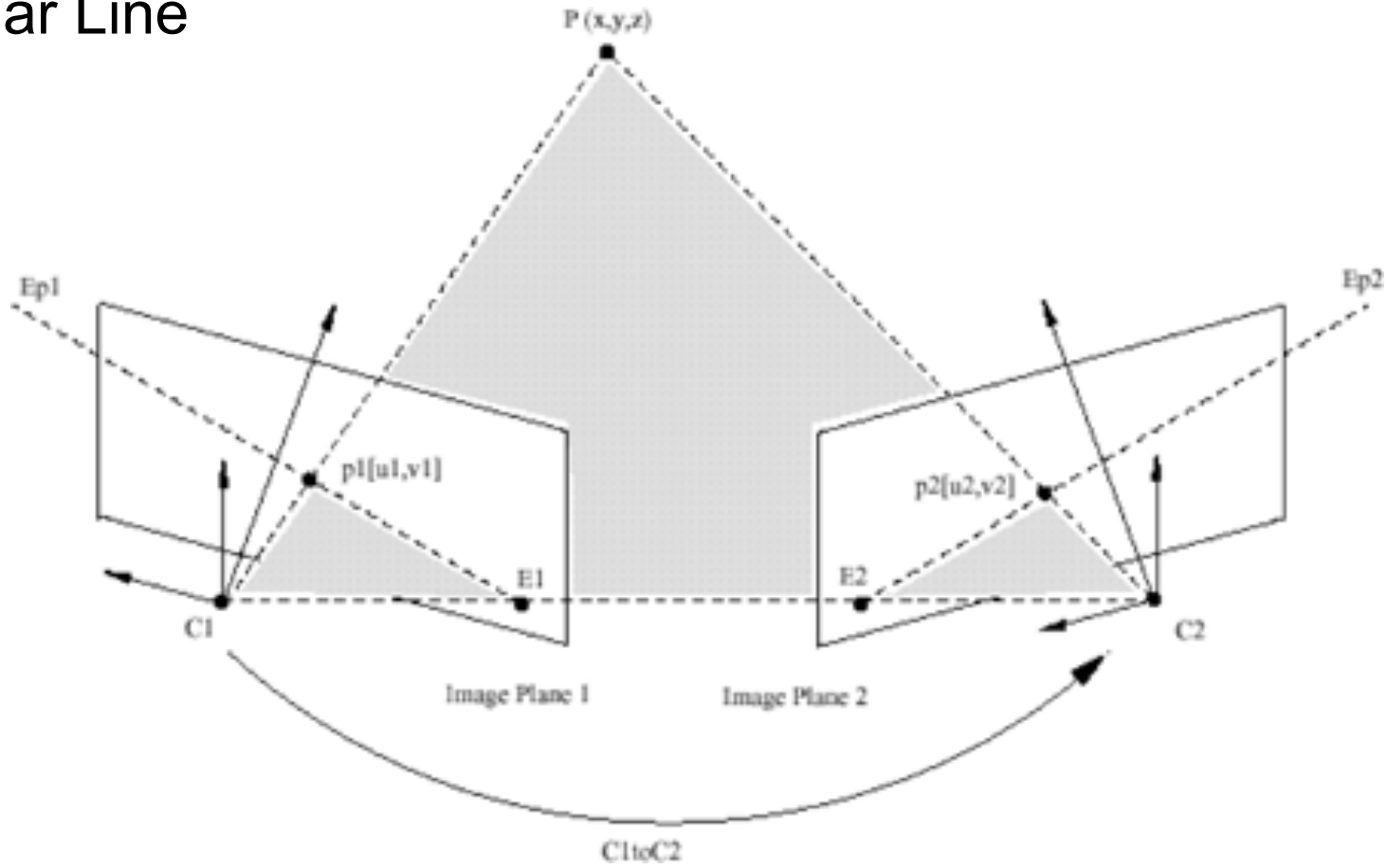
Stereo Vision: Correspondence Problem

- Matching between points in the two images which are projection of the same 3D real point
- Correspondence search could be done by comparing the observed points with all other points in the other image. Typical similarity measures are the Correlation and image Difference.
- This image search can be computationally very expensive! Is there a way to make the correspondence search 1 dimensional?



Correspondence Problem: Epipolar Constraint

- The correspondent of a point in an image must lie on a line in the other image, called Epipolar Line

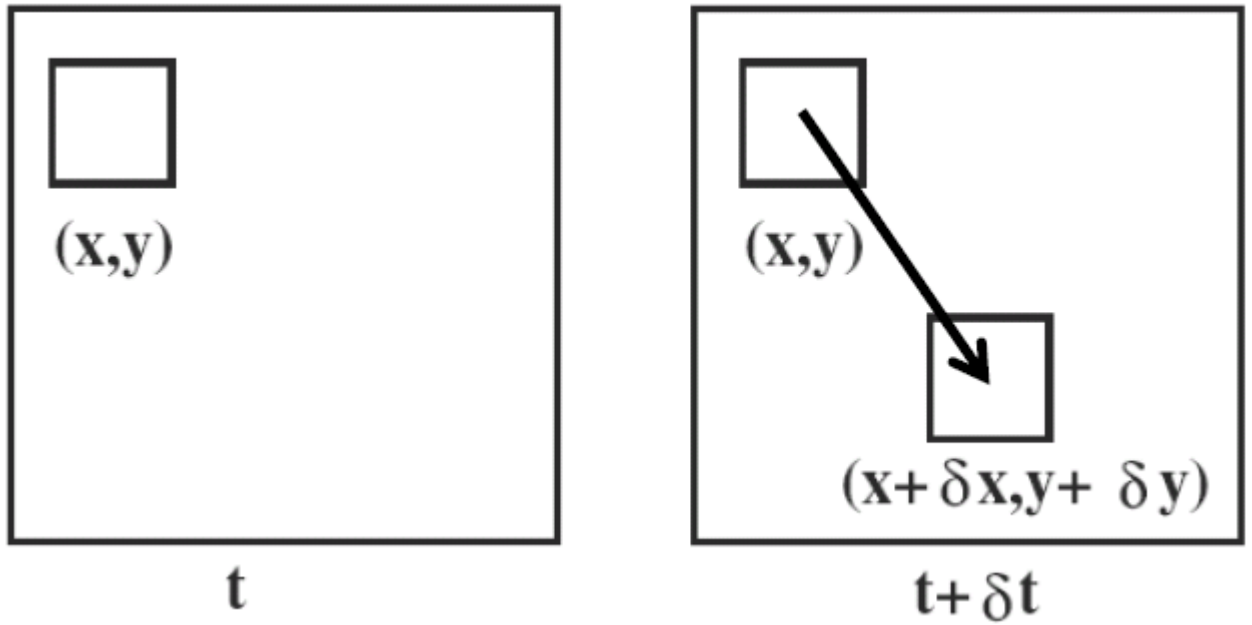


Correspondence Problem: Epipolar Constraint

- Thanks to the epipolar constraint, conjugate points can be searched along epipolar lines: this reduces the computational cost to 1 dimension!

Optical Flow

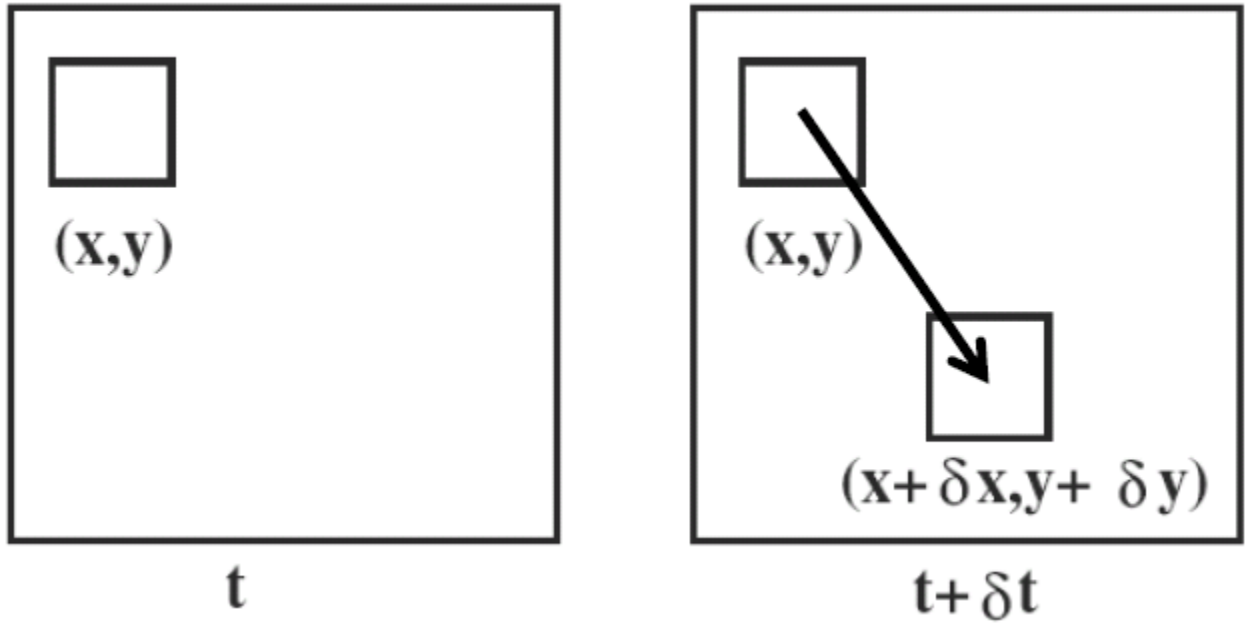
- It computes the motion vectors of all pixels in the image (or a subset of them to be faster)



$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$$

- Applications include collision avoidance

Optical Flow



$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0$$

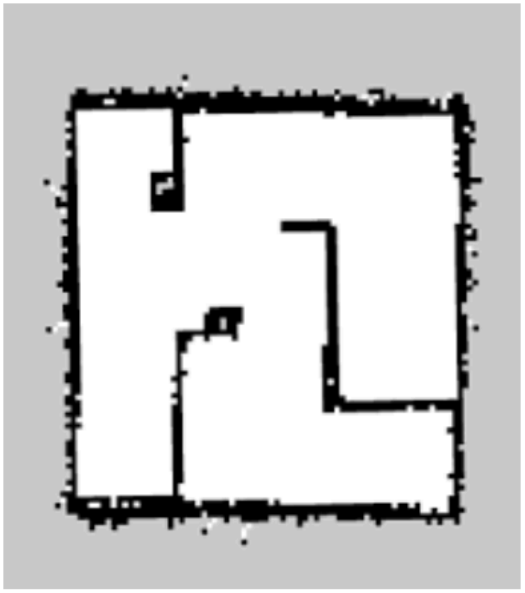
$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t}\frac{\delta t}{\delta t} = 0$$

$$\frac{\partial I}{\partial x}v_x + \frac{\partial I}{\partial y}v_y + \frac{\partial I}{\partial t} = 0$$

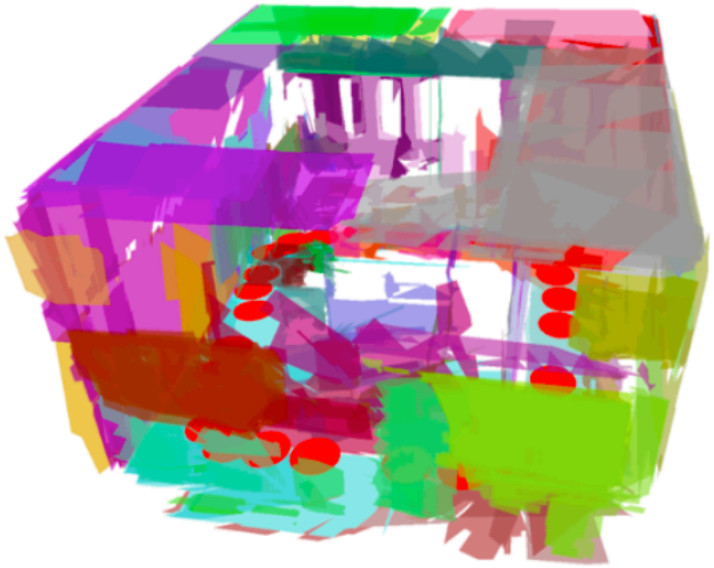
$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$$

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + \cdots$$

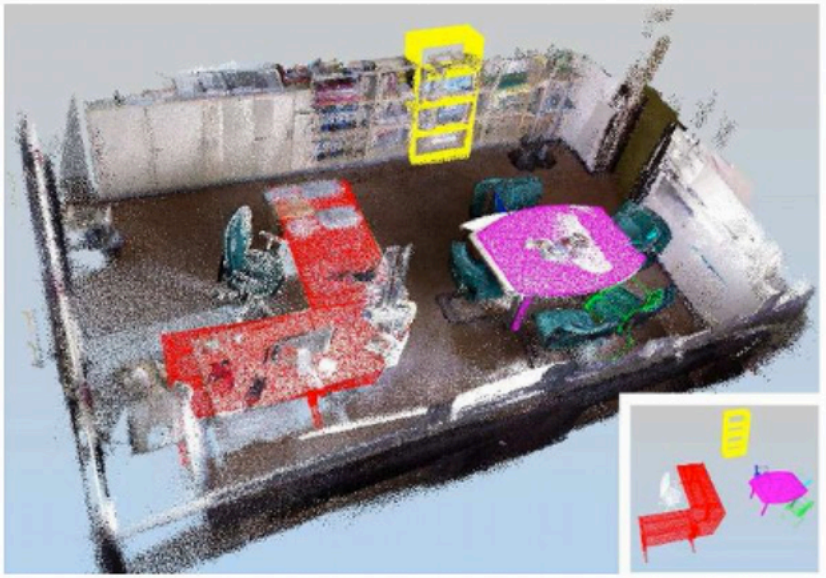
80% wrong: Basic assumption Optical Flow: Brightness constancy constraint



(a)

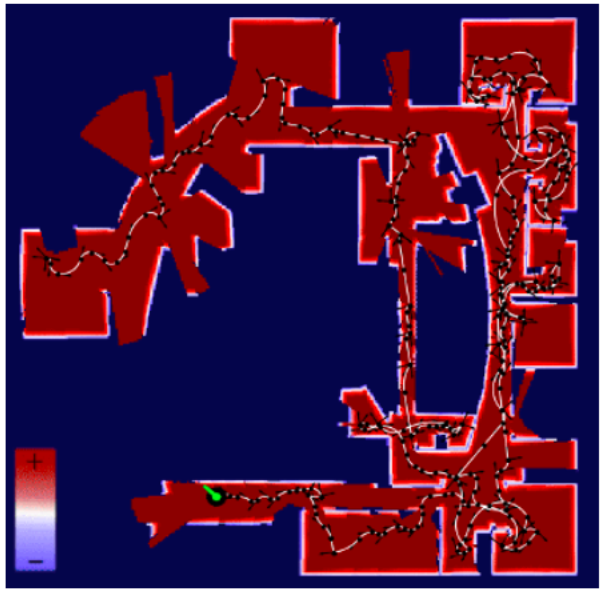


(b)



(c)

(d)



(e)

(f)

SLAM Front-end & Back-end

- Front-end
  - calculate relative poses between several frames/ to map
    - scan matching
    - image registration
    - …
  - estimate absolute poses
  - construct the local map
- Back-end
  - optimize the absolute poses and mapping
  - only if a loop was closed

https://ww2.mathworks.cn/help/nav/ug/implement-simultaneous-localization-and-mapping-with-lidar-scans.html

THREE SLAM PARADIGMS

The Three SLAM Paradigms

- Most of the SLAM algorithms are based on the following three different approaches:
  - Extended Kalman Filter SLAM: (called EKF SLAM): parametric representation of uncertainties
  - Particle Filter SLAM: (called FAST SLAM): non-parametric representation of uncertainties
  - Graph-Based SLAM

EKF SLAM: overview

- **Extended state vector** y_t : robot pose x_t + position of all the features m_i in the map:

$$y_t = [x_t, m_0, \ldots, m_{n-1}]^T$$

- Example: 2D line-landmarks, size of y_t = $3+2n$: three variables to represent the robot pose + $2n$ variables for the n line-landmarks having vector components

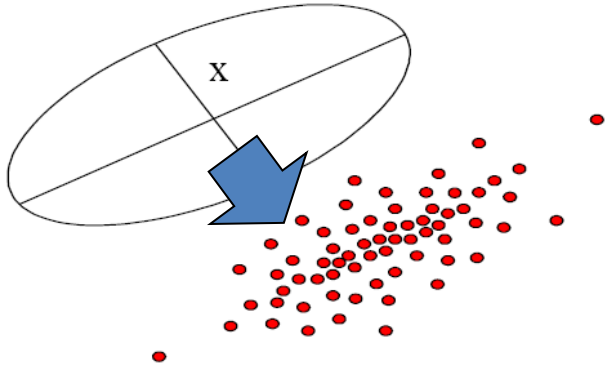
$$(\alpha_i, r_i)$$

$$y_t = [x_t, y_t, \theta_t, \alpha_0, r_0, \ldots, \alpha_{n-1}, r_{n-1}]^T$$

- As the robot moves and takes measurements, the state vector and covariance matrix are updated using the standard equations of the extended Kalman filter
- Drawback: EKF SLAM is computationally very expensive.

Particle Filter SLAM: FastSLAM

- **FastSLAM approach**
  - Using particle filters.
  - Particle filters: mathematical models that represent probability distribution as a set of discrete particles that occupy the state space.



probability distribution (ellipse) as particle set (red dots)

- Particle filter update
  - Generate new particle distribution using motion model and controls

a) For each particle:

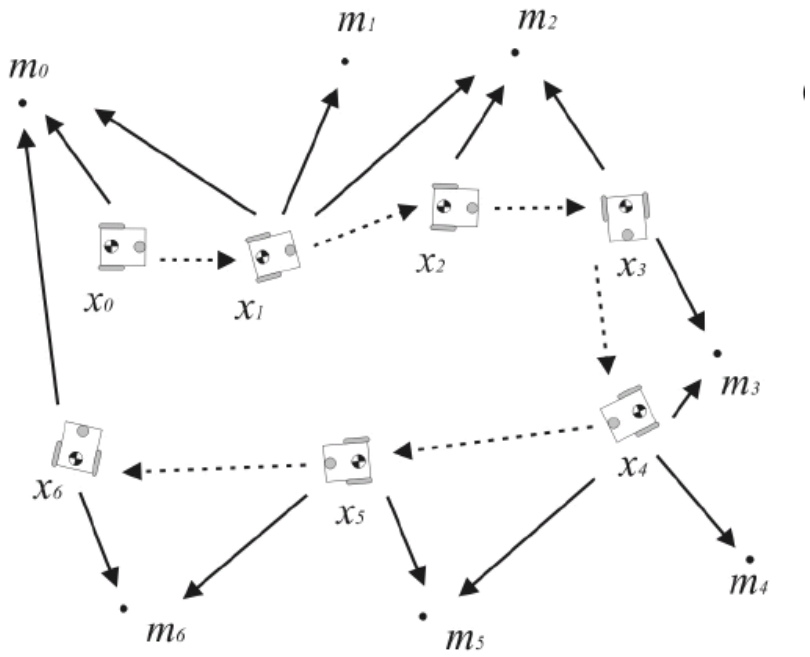
1. Compare particle's prediction of measurements with actual measurements
2. Particles whose predictions match the measurements are given a high weight

b) Filter resample:

- Resample particles based on weight
- Filter resample
  - Assign each particle a weight depending on how well its estimate of the state agrees with the measurements and randomly draw particles from previous distribution based on weights creating a new distribution.

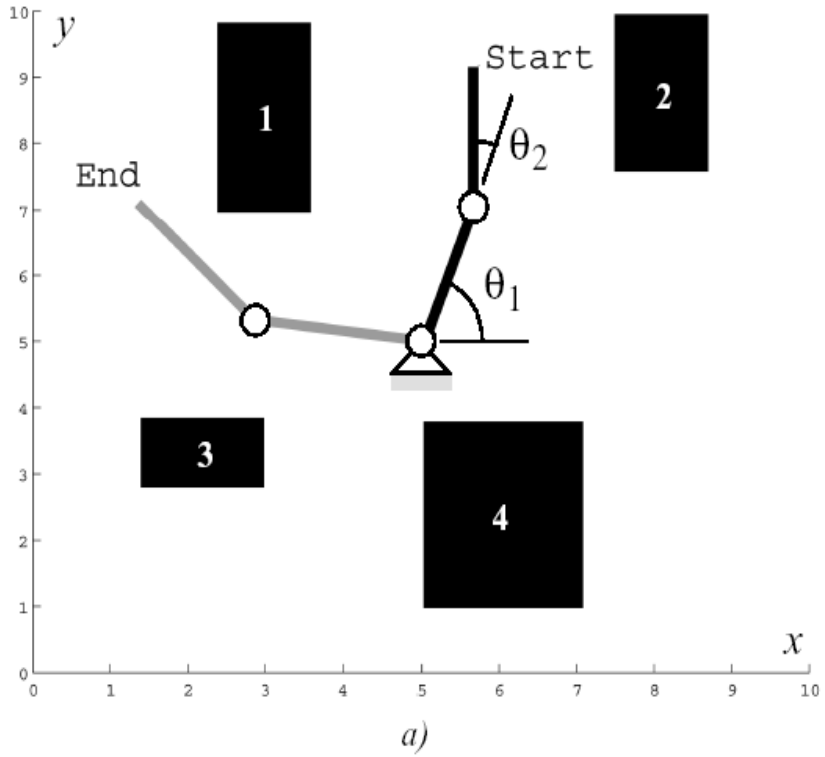
Graph-Based SLAM (1/3)

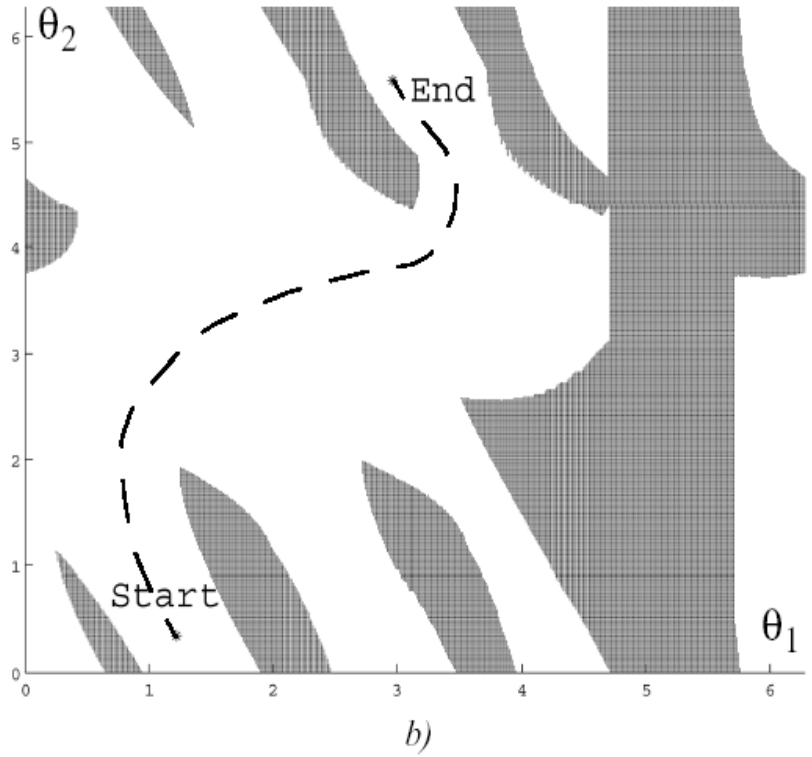
- SLAM problem can be interpreted as a sparse graph of nodes and constraints between nodes.
- The nodes of the graph are the robot locations and the features in the map.
- Constraints: relative position between consecutive robot poses , (given by the odometry input ***u***) and the relative position between the robot locations and the features observed from those locations.



Work Space (Map) → Configuration Space

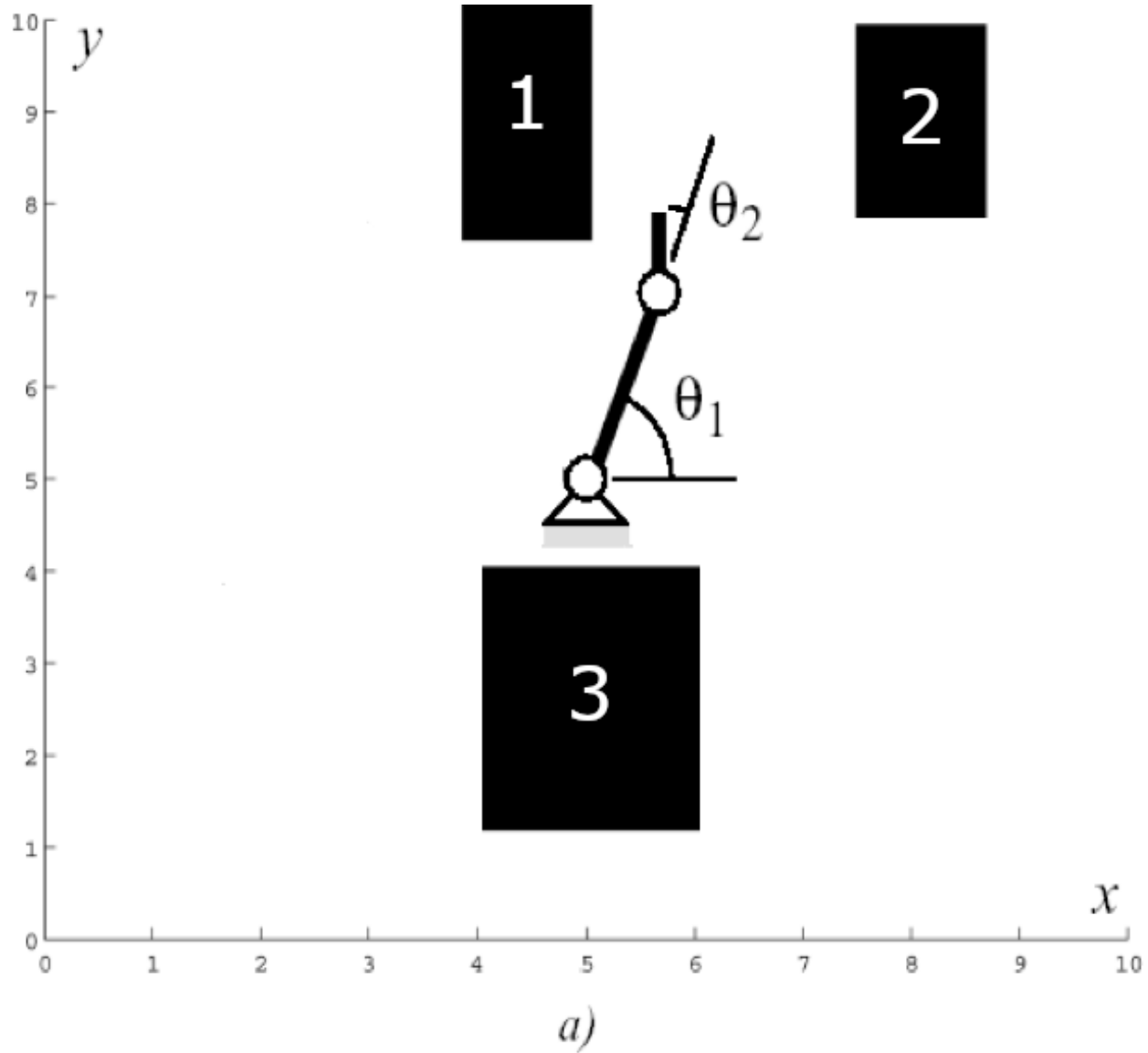
- State or configuration q can be described with k values q_i



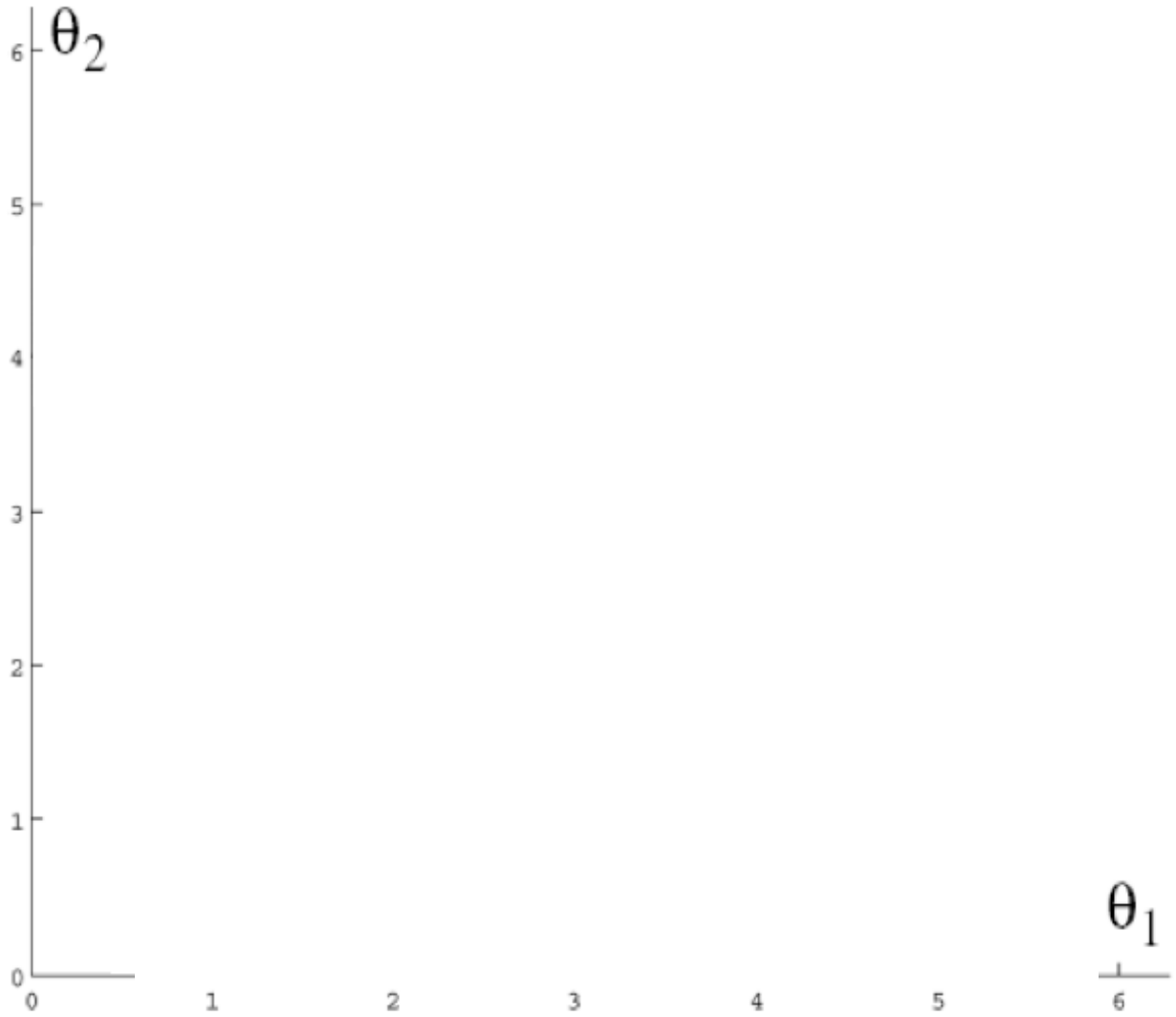


Work Space
Dimension depends on map
Dimension – typically 2D or 3D

Configuration Space:
the dimension of this
space is equal to the Degrees of Freedom (DoF)
of the robot

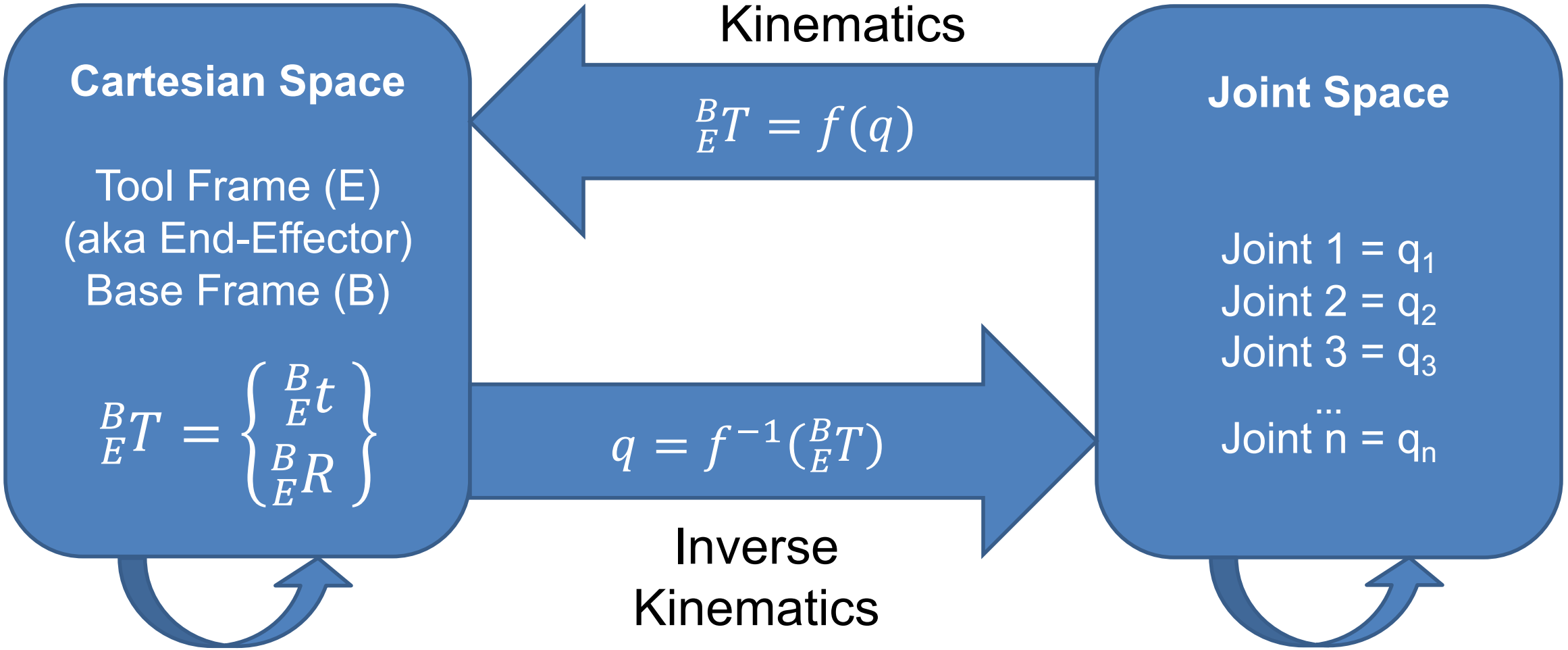


a)



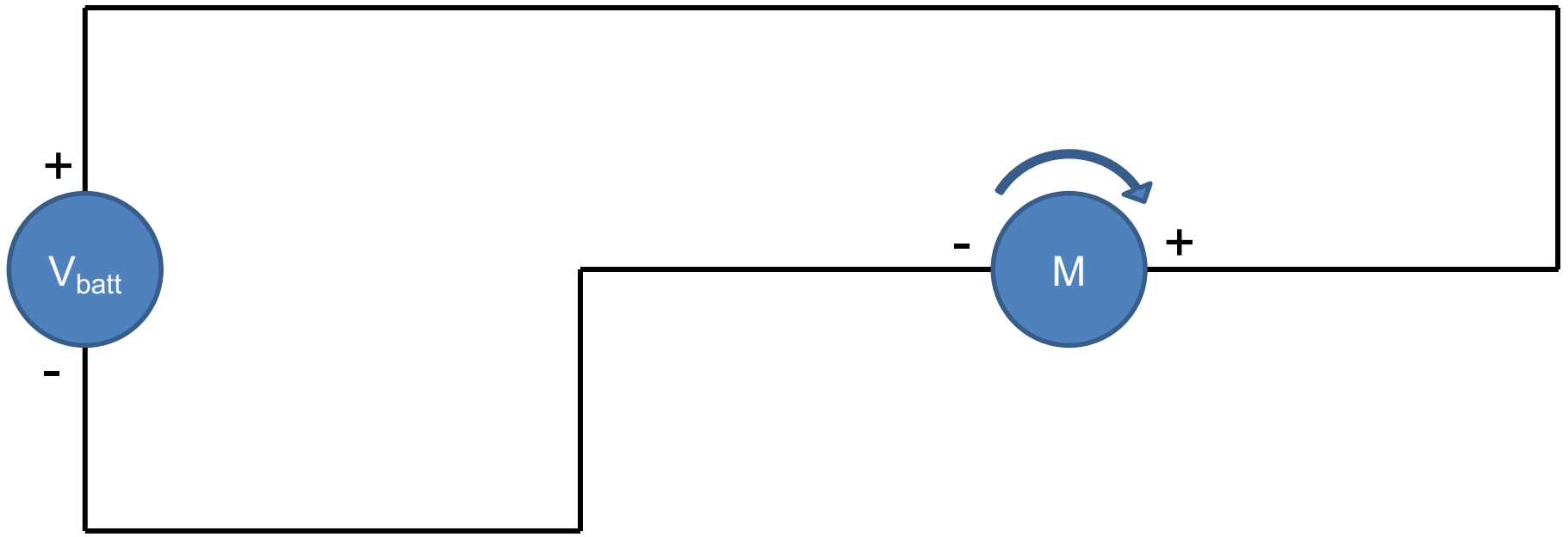
b)

Kinematics


Forward
Kinematics
$^{B}_{E}T = f(q)$
Cartesian Space
Tool Frame (E)
(aka End-Effector)
Base Frame (B)
$^{B}_{E}T = \begin{Bmatrix} ^{B}_{E}t \\ ^{B}_{E}R \end{Bmatrix}$
Joint Space
Joint 1 = q_1
Joint 2 = q_2
Joint 3 = q_3
...
Joint n = q_n
$q = f^{-1}(^{B}_{E}T)$
Inverse
Kinematics
Rigid body transformation
Between coordinate frames
Linear algebra

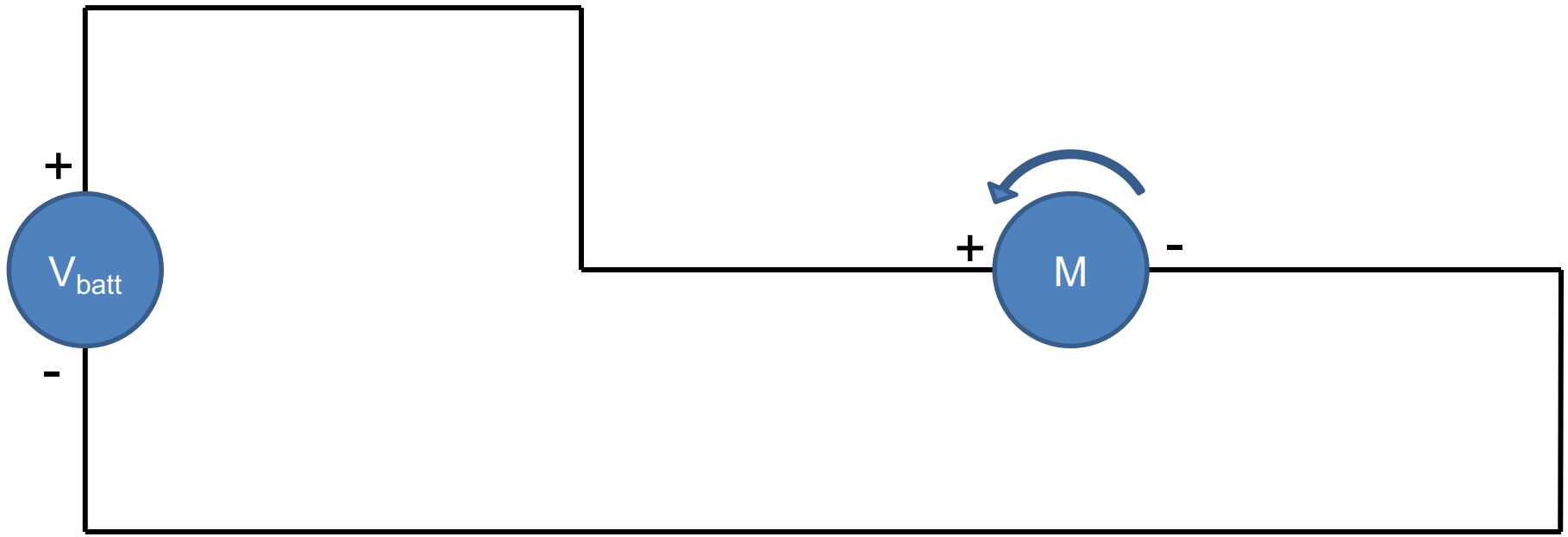
How does a Motor Driver work? H-Bridge

- H-Bridge:
  - Change direction of energy flow -> change direction of motor
  - Also: switch motor on and off



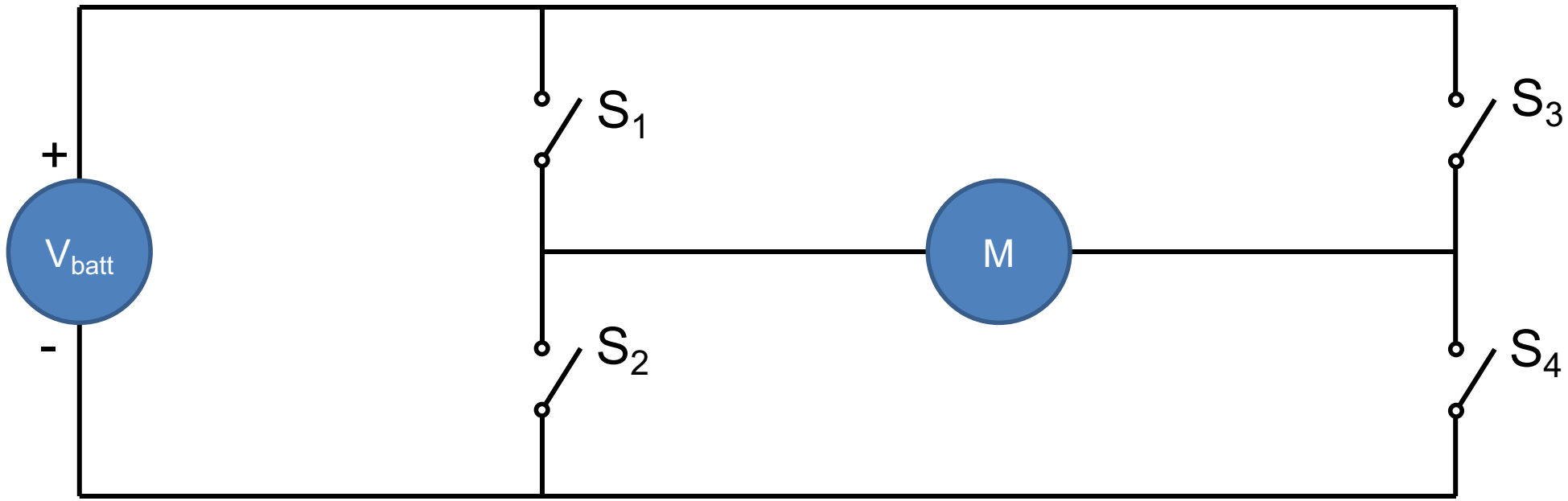
How does a Motor Driver work? H-Bridge

- H-Bridge:
  - Change direction of energy flow -> change direction of motor
  - Also: switch motor on and off



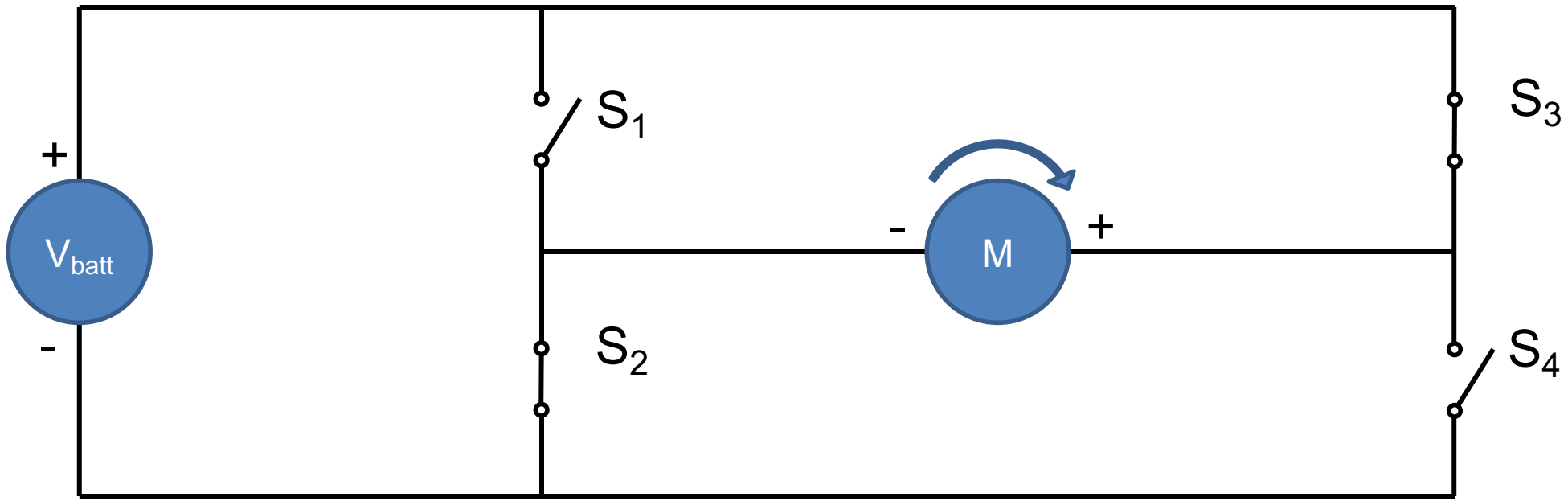
How does a Motor Driver work? H-Bridge

- H-Bridge:
  - Change direction of energy flow -> change direction of motor
  - Also: switch motor on and off



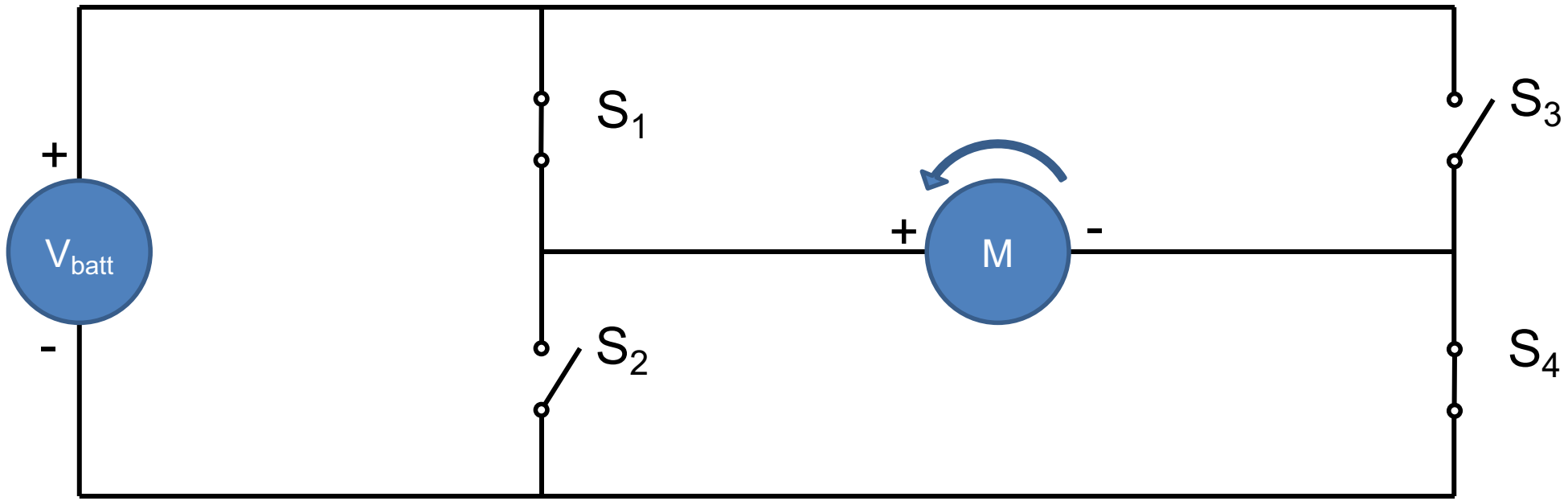
How does a Motor Driver work? H-Bridge

- H-Bridge:
  - Change direction of energy flow -> change direction of motor
  - Also: switch motor on and off



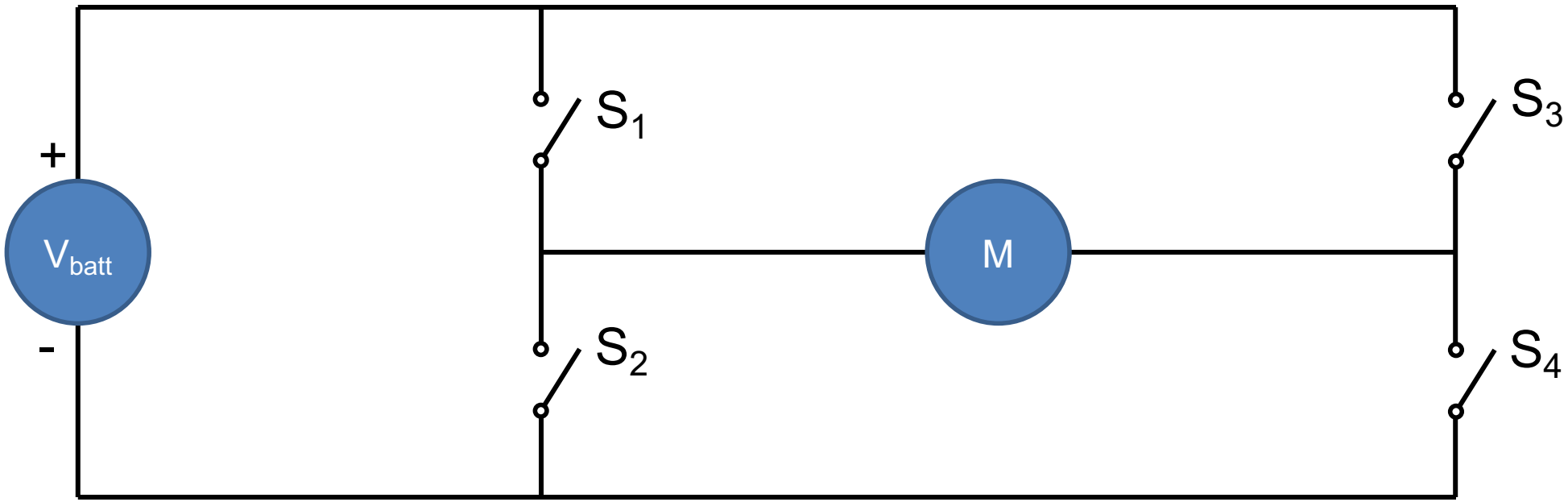
How does a Motor Driver work? H-Bridge

- H-Bridge:
  - Change direction of energy flow -> change direction of motor
  - Also: switch motor on and off



How does a Motor Driver work? H-Bridge

- Short-cut: burn switches/ wires – Motor should be OK…
  - S_1 and S_2 or
  - S_3 and S_4



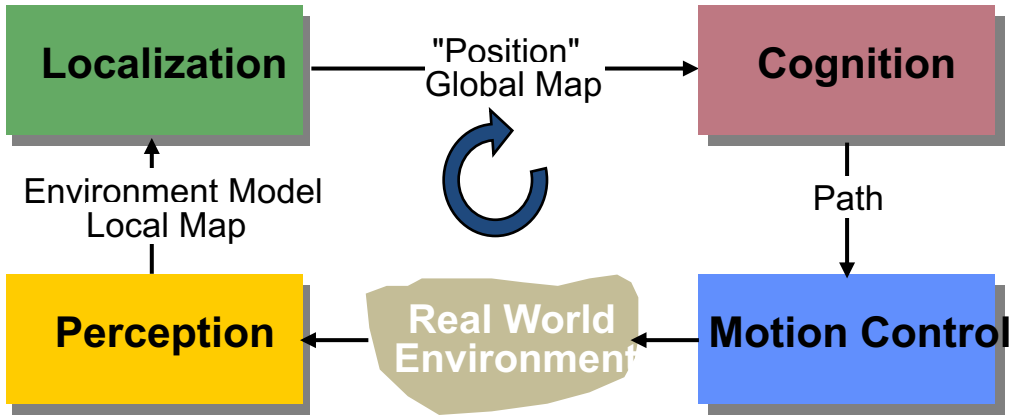
Q & A

BEHAVIOR BASED ROBOTICS

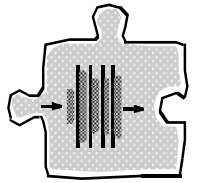
"Small Model" – "Little Brain"

Control Architectures / Strategies

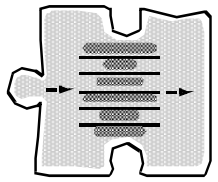
- Control Loop
  - **dynamically changing**
  - **no compact model available**
  - **many sources of uncertainty**



- Three Approaches
  - Classical AI (Big Model)
    - complete modeling
    - function based
    - horizontal decomposition



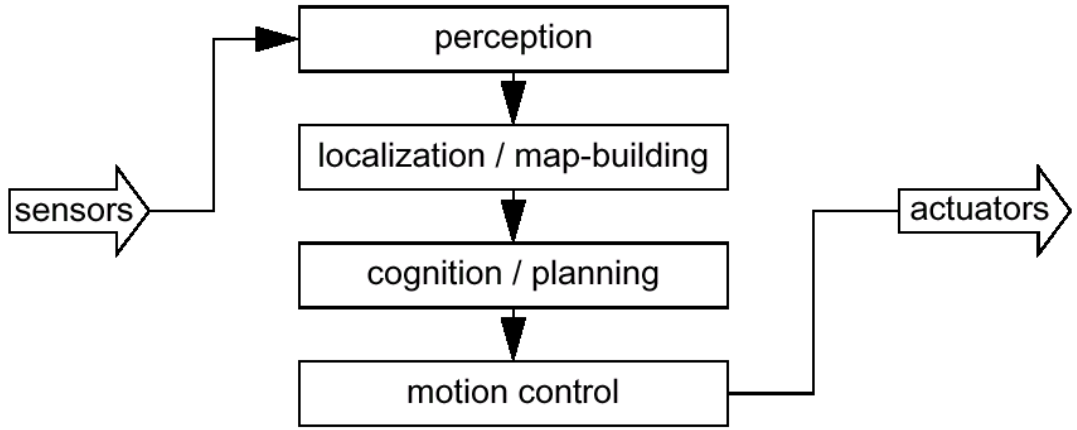
  - New AI (Nouvelle AI; Small Model; Behavior Based Robotics)
    - sparse or no modeling
    - behavior based
    - vertical decomposition
    - bottom up

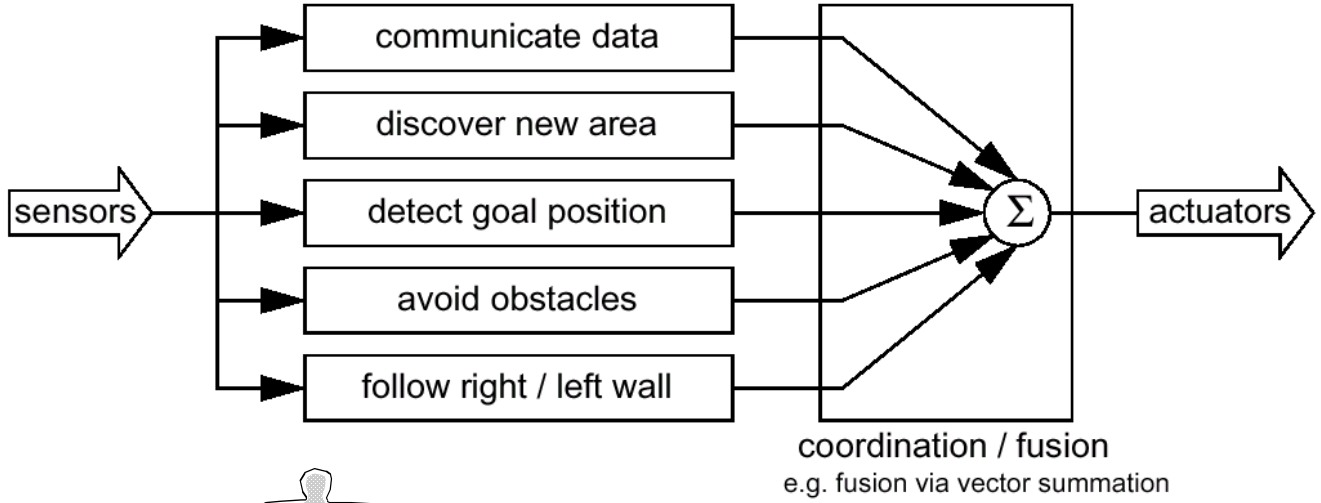


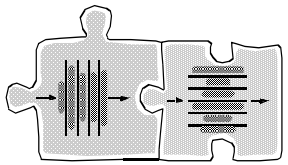
  - DL/ Reinforcement Learning

Two Approaches

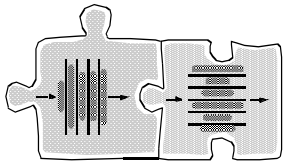
- Classical AI
  (model based navigation)
  - complete modeling
  - function based
  - horizontal decomposition
- New AI
  (behavior based navigation)
  - sparse or no modeling
  - behavior based
  - vertical decomposition
  - bottom up
- Possible Solution
  - Combine Approaches (= Hybrid Approach)

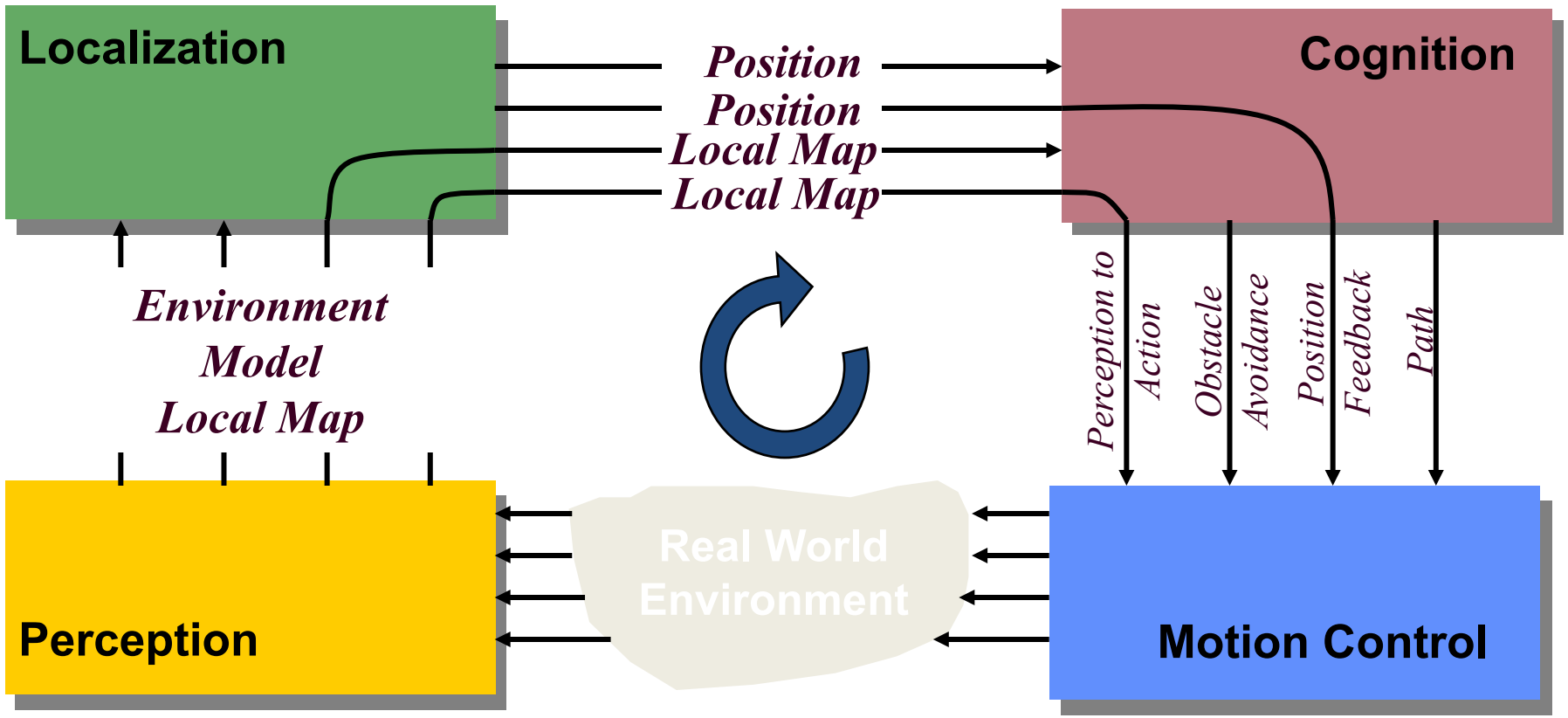






Mixed Approach Depicted into the General Control Scheme



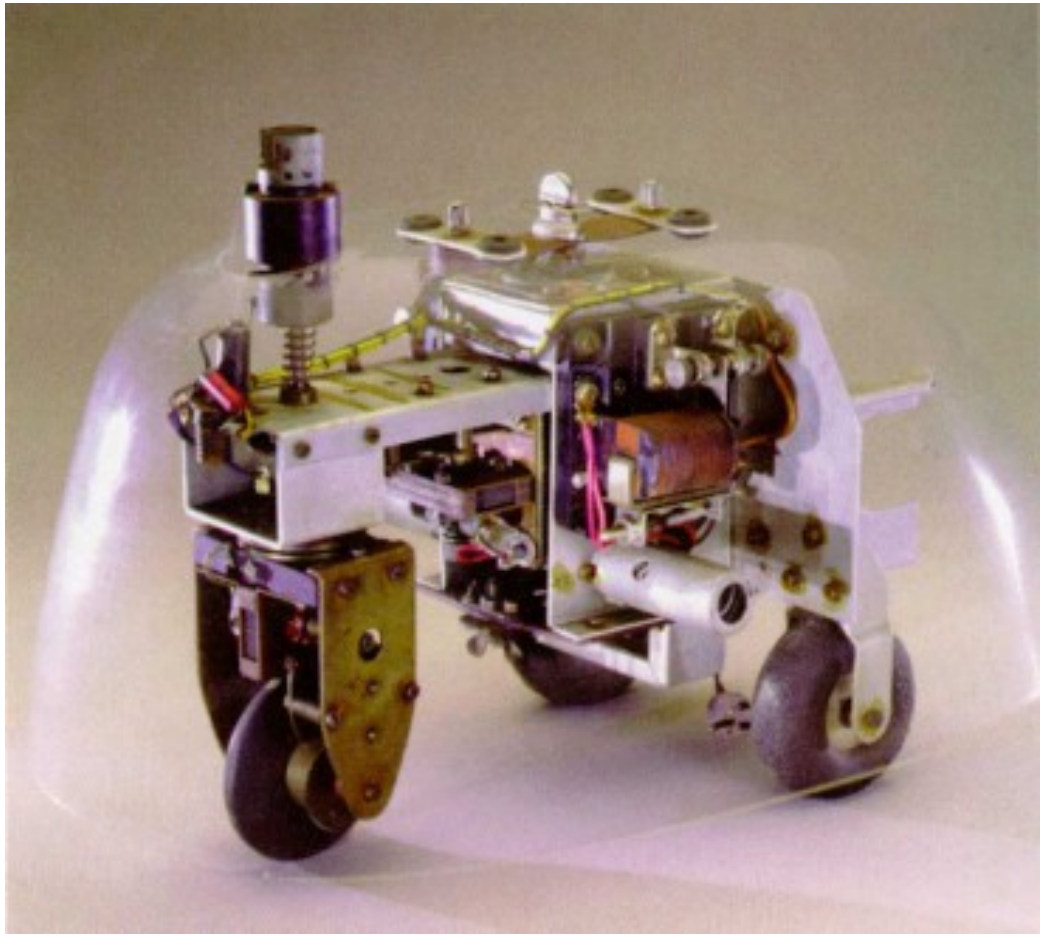


Emergence

- Adaptive behavior
  - emerges from complex interactions between body, world and brain
- Non-centrally controlled (or designed) behavior
  - results from the interactions of multiple simple components
- Meanings:
  - Surprising situations or behaviors
  - Property of system not contained in any of its parts
  - Behavior resulting from agent-environment interaction not explicitly programmed
- Ant colony:
  - self-organized; simple individuals; local interactions =>
  - emergent behavior - No global control

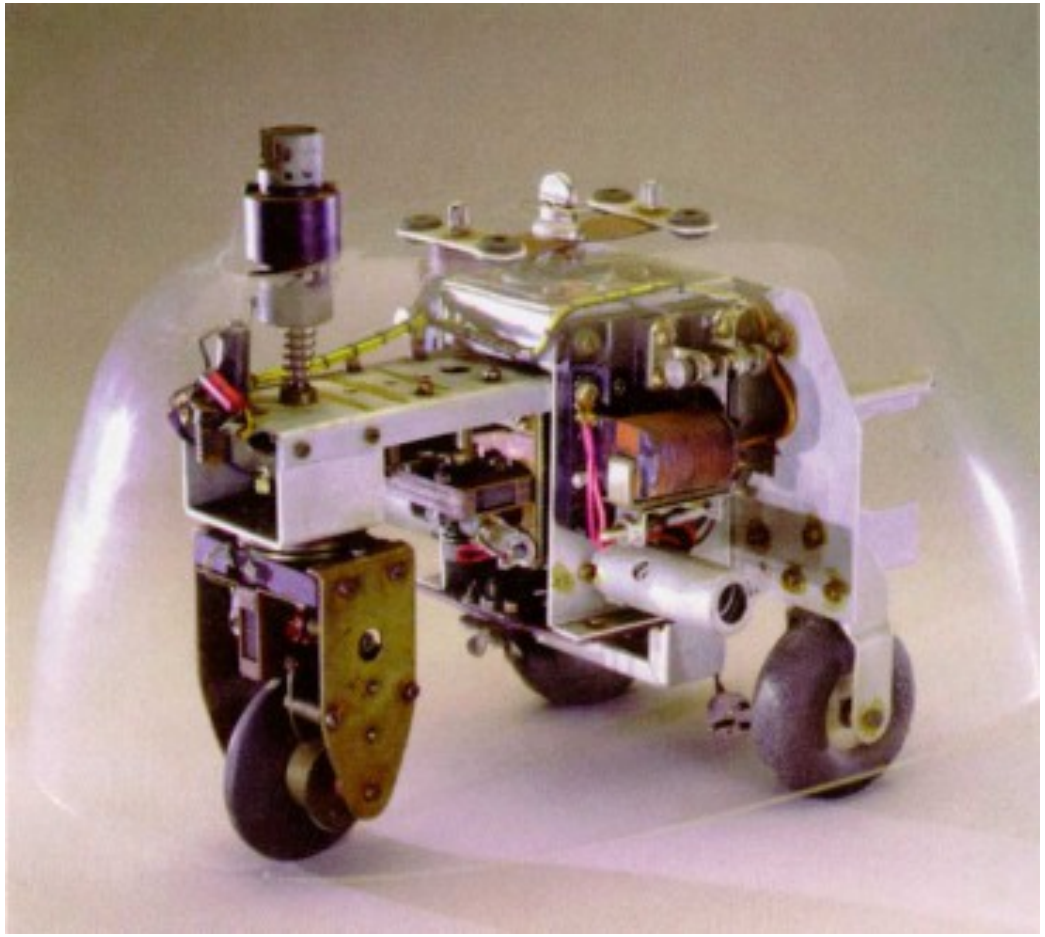
Grey Walter's Tortoise

- Turtle shape robots 1949
- Purely analogue electronics
- Phototaxis: go towards the light
- Sensors:
  - 1 photocell,
  - 1 bump sensor
- 2 motors
- Reactive control



Grey Walter's Tortoise

- Behaviors:
  - Seek light
  - Head toward weak light
  - Back away from bright light
  - Turn and push (obstacle avoidance)
  - Recharge battery

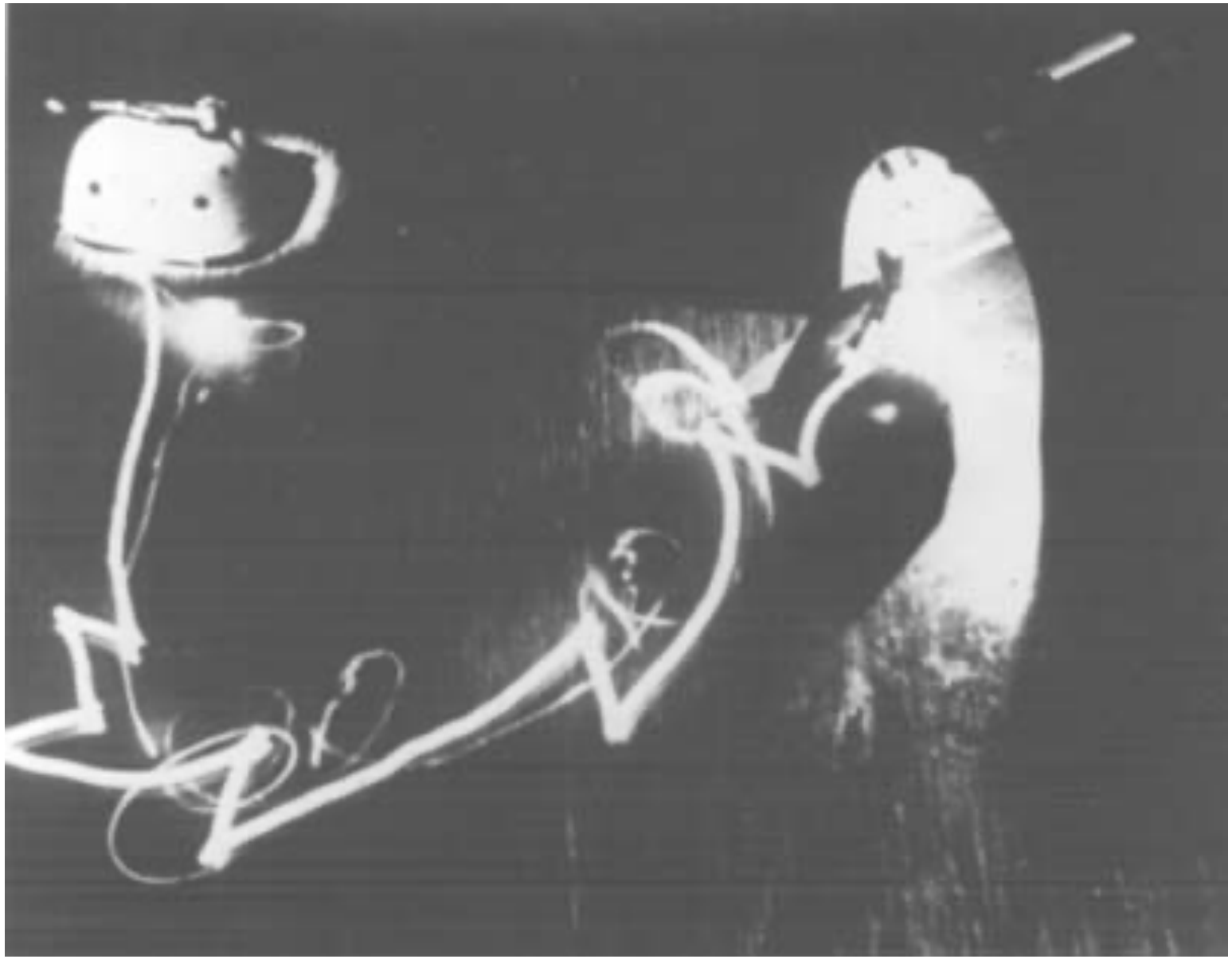


Turtle Principles

- Simple is better
  - e.g., clever recharging strategy
- Exploration/ speculation: keeps moving
  - except when charging
- Attraction:
  - motivation to approach light
- Aversion:
  - motivation to avoid obstacles, slopes

Tortoise behavior

- A path: a candle on top of the shell

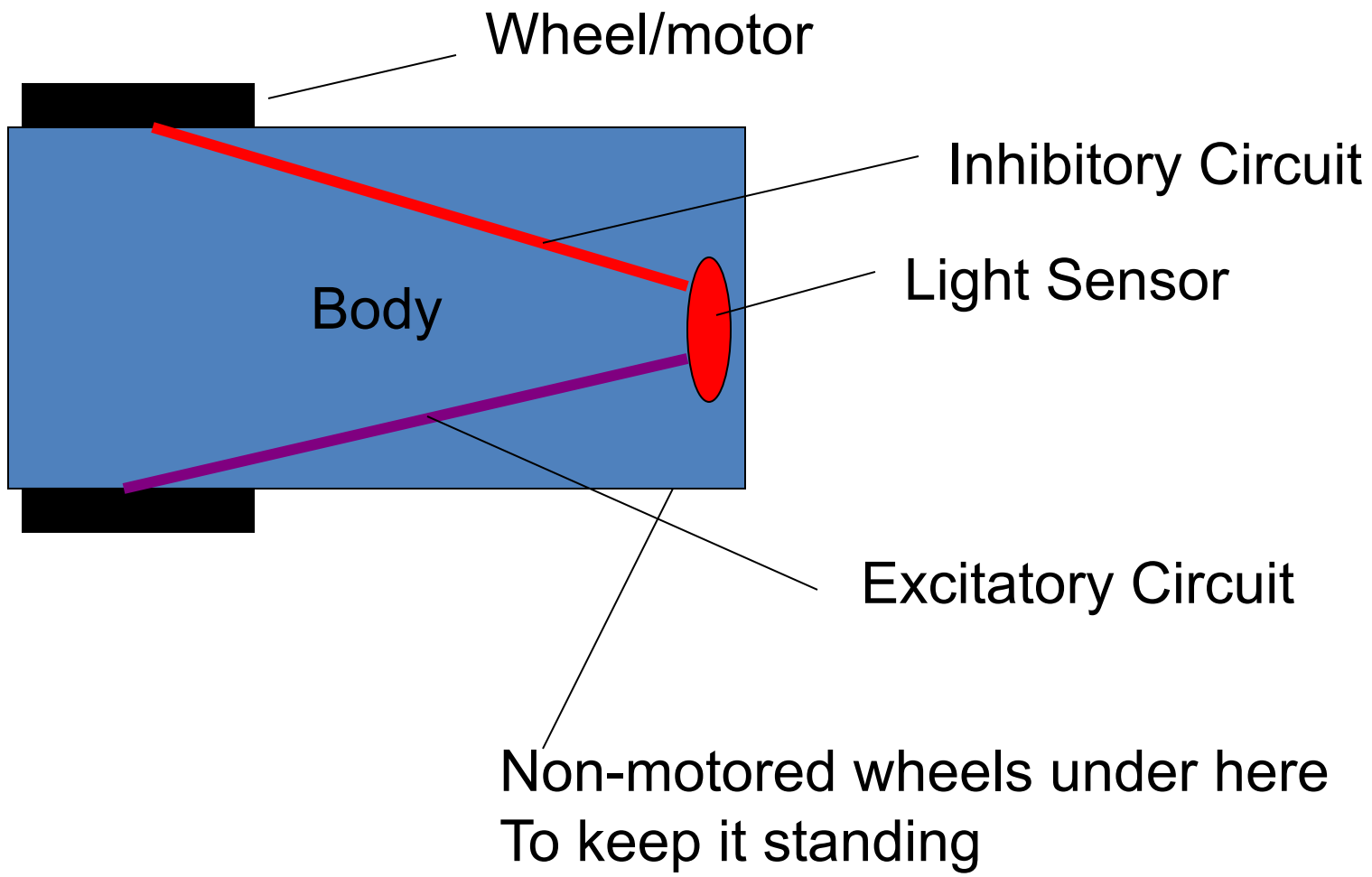


Video …


BRITISH
PATHÉ

Braitenberg's Vehicles

- Valentino Braitenberg (1926)
- 1984: "Vehicles: Experiments in Synthetic Psychology"

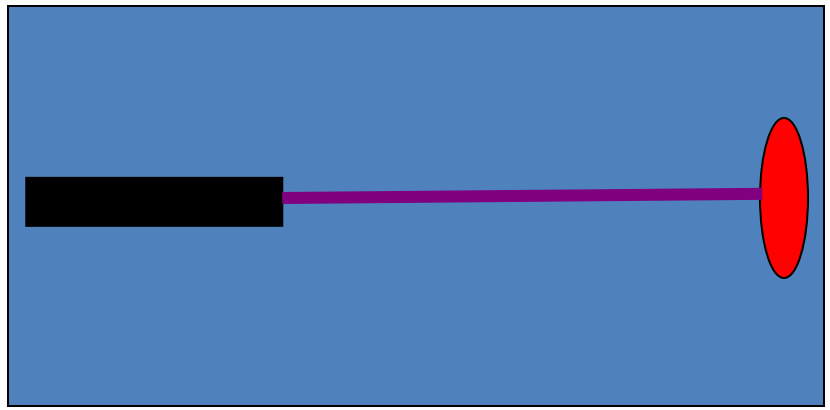


Material: Tommy Walsh http://my.fit.edu/~twalsh/Braitenberg%20Vehicles.ppt

Definitions

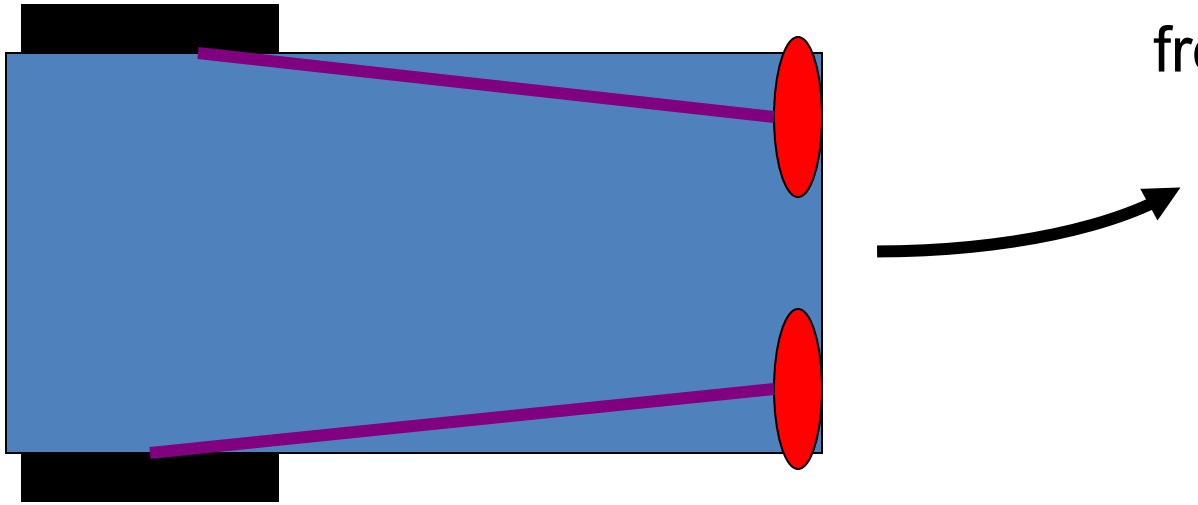
- Inhibitory circuit: when sensor gets activated, motor slows
- Excitatory circuit: when sensor gets activated, motor speeds
- Sensor is a light sensor, unless otherwise noted

Vehicle 1: Alive



Basic Braitenberg vehicle:
Goes towards light source

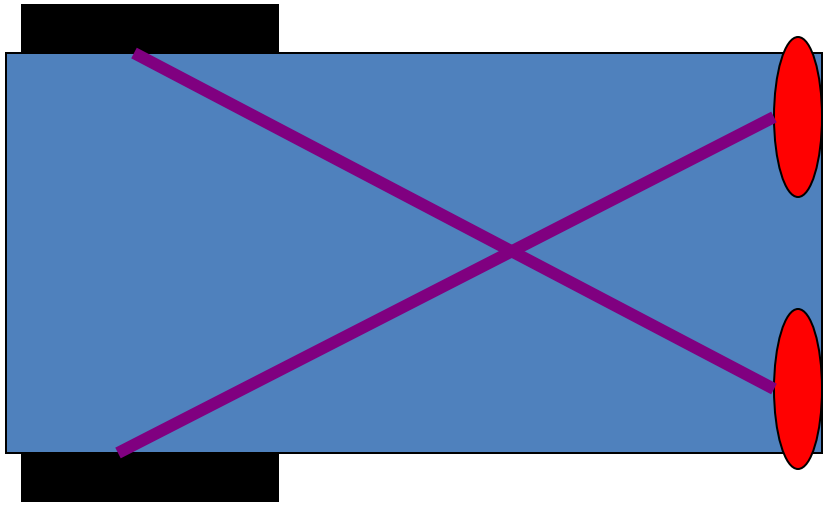
Vehicle 2: Cowardly



More light right →
right wheel turns faster →
turns towards the left, away from the light.

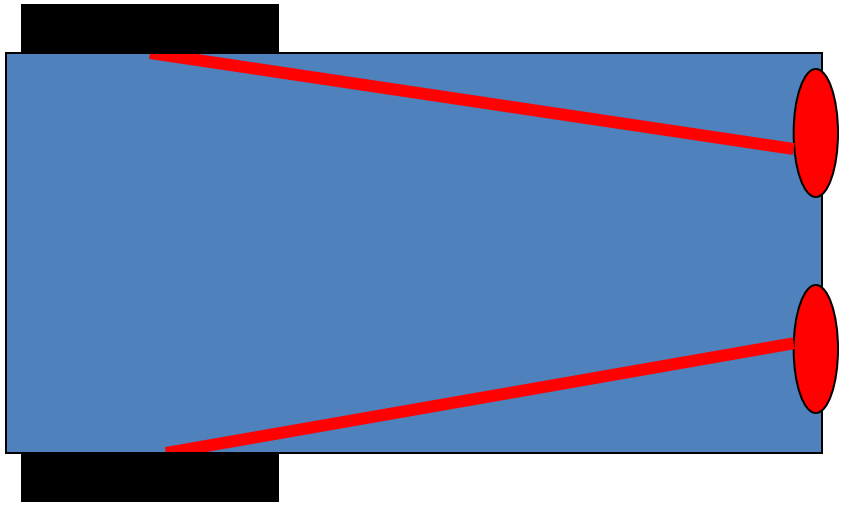
Demonstrates “fight or flight” instinct in animals
Turns away from light if one sensor is activated more than the other
If both are equal, light source is “attacked”

Vehicle 2b: Aggressive



Faces light source and drives toward it

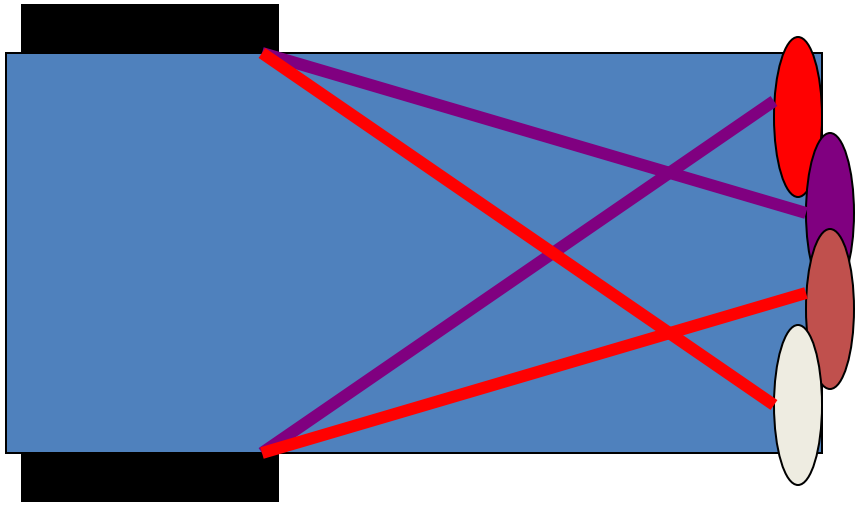
Vehicle 3: Loving



Drives forward
Faces the light source and slows down

Models love/adoration

A little more complicated: Vehicle 3c: Knowing

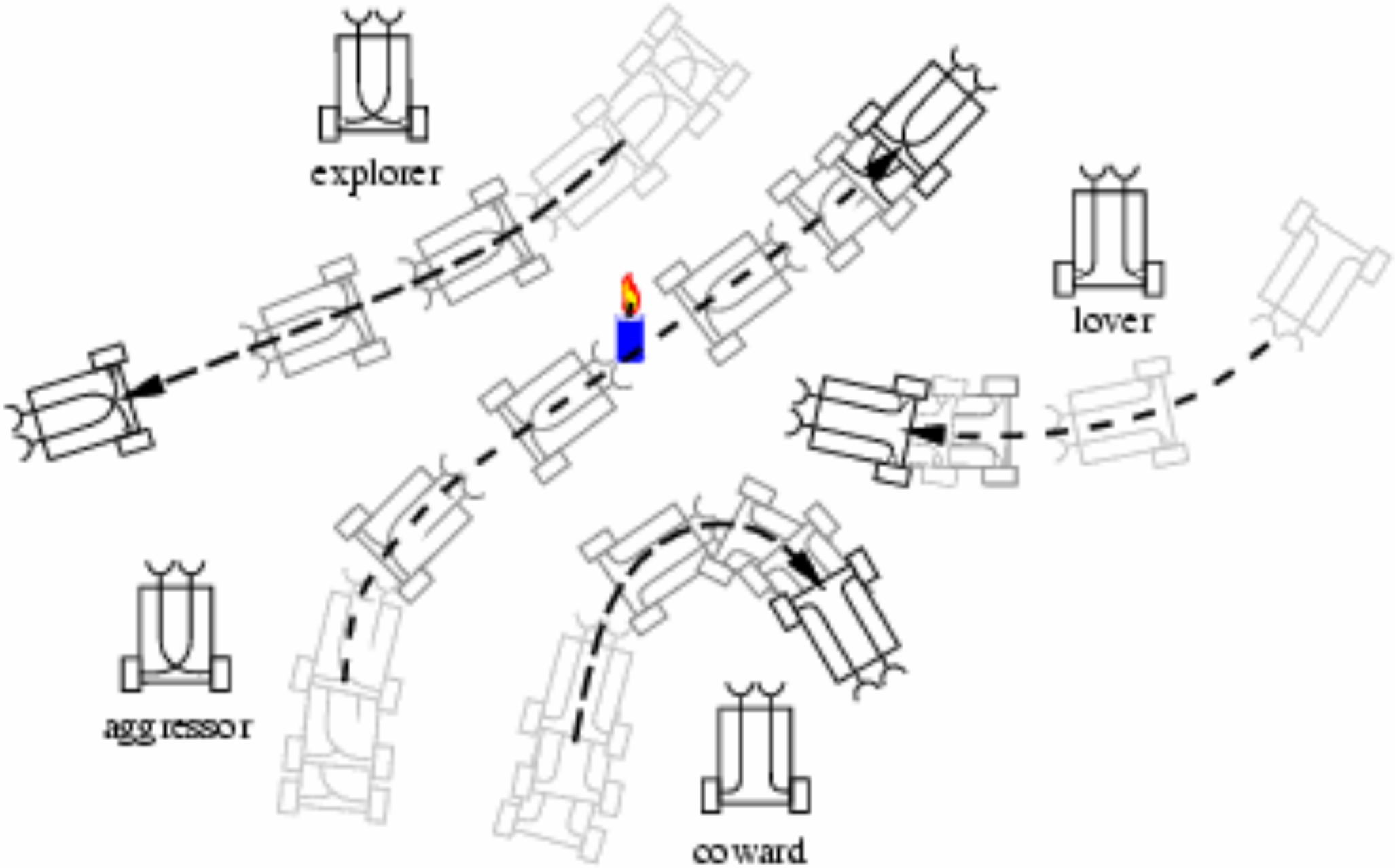


Light Sensor
Temperature Sensor
Organic Material Sensor
Oxygen Sensor

Different sensors:

Turns towards light,
doesn’t like heat,
loves organic material,
searches for best Oxygen

Emergent Behavior: Performs the brain function of simplest living beings


explorer
lover
aggressor
coward

Subsumption architecture

- Rodney Brooks (MIT; founder of iRobot and Rethink Robotics) 1991:
  - “The world is its own best model” =>
  - “Intelligence without representation”
- Emergent behaviors

- Conclusions:
  - Emergent behaviors quite interesting/ impressive.
  - More complex tasks often need more intelligence.
  - => Behaviors good for low level tasks.