

# Fundamental Problems In Robotics

- What does the world look like? (**mapping**)
  - sense from various positions
  - integrate measurements to produce map
  - assumes perfect knowledge of position
- Where am I in the world? (**localization**)
  - Sense
  - relate sensor readings to a world model
  - compute location relative to model
  - assumes a perfect world model
- Together, these are **SLAM** (Simultaneous Localization and Mapping)

# Sensors

- **Proprioceptive Sensors**

(monitor state of robot)

- IMU (accels & gyros)
- Wheel encoders
- Doppler radar ...



- **Exteroceptive Sensors**

(monitor environment)

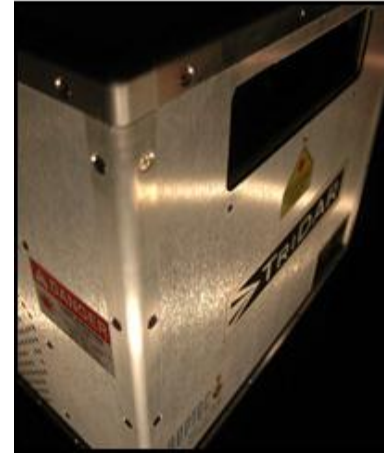
- Cameras (single, stereo, omni, FLIR ...)
- Laser scanner
- MW radar
- Sonar
- Tactile...



# Types of sensor

## Specific examples

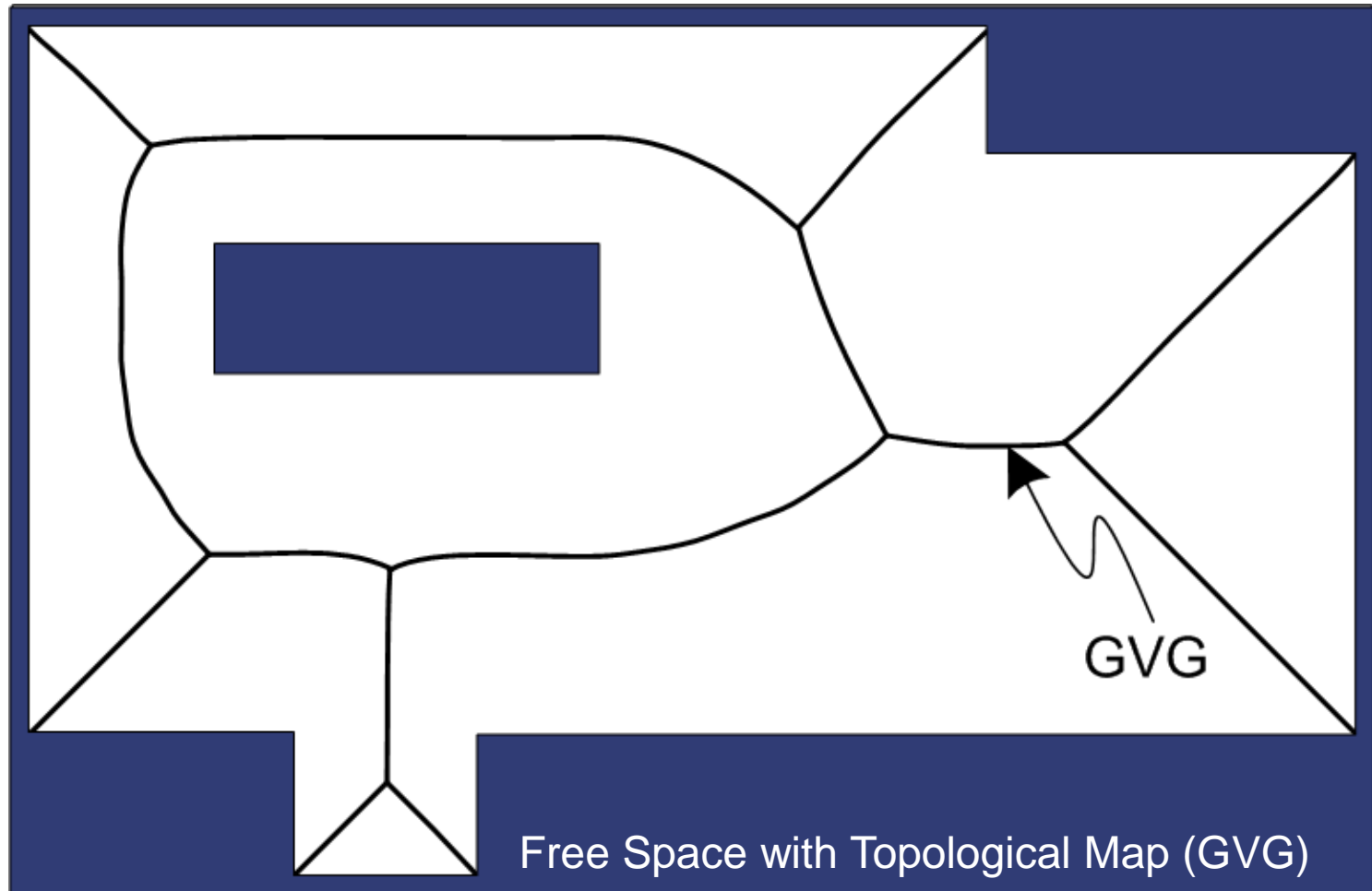
- tactile
- close-range proximity
- angular position
- infrared
- Sonar
- laser (various types)
- radar
- compasses, gyroscopes
- Force
- GPS
- vision



# Path Planning

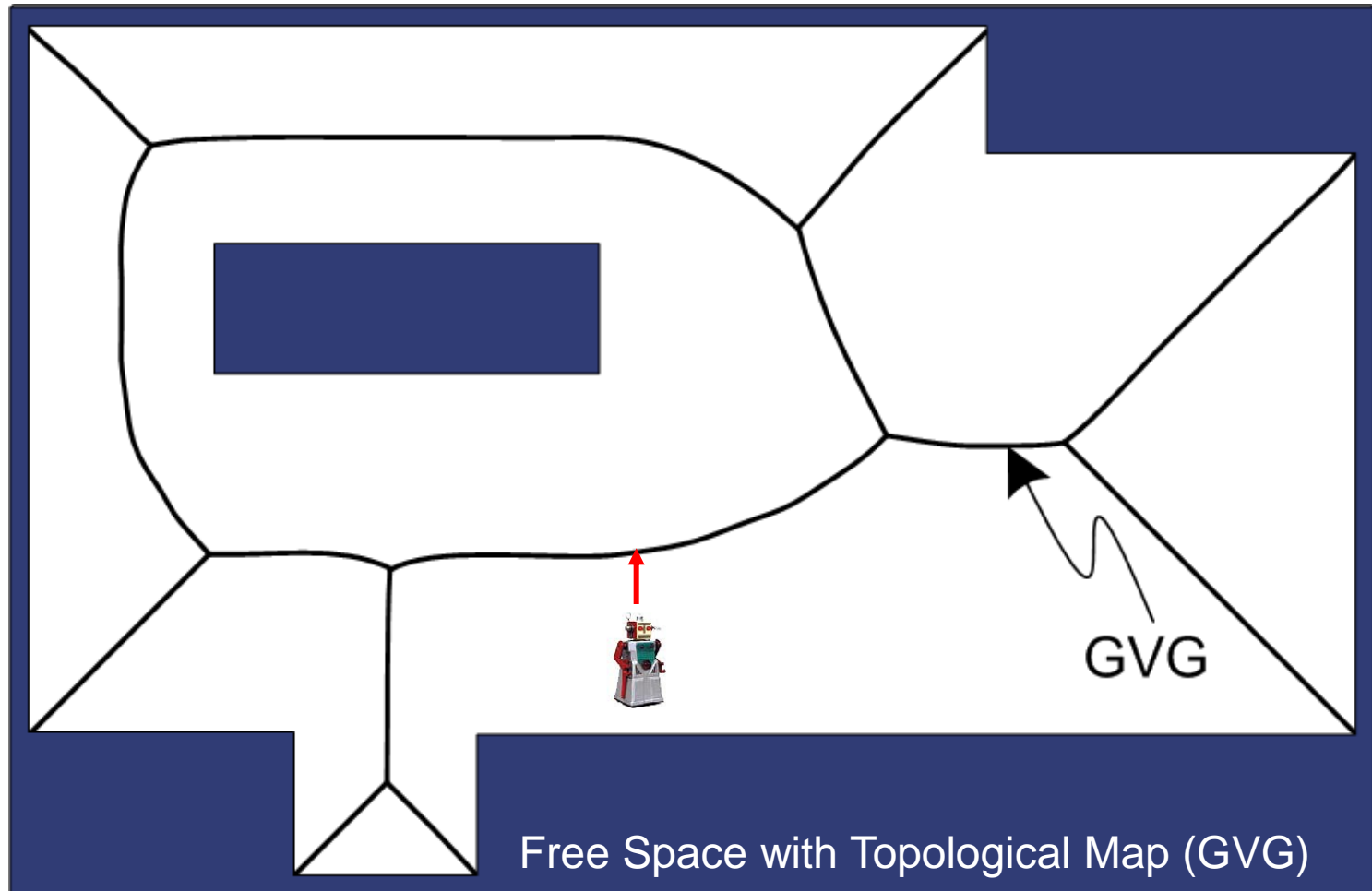
- Visibility Graph
- Bug Algorithms
- Potential Fields
- Skeletons/Voronoi Graphs
- C-Space
- PRM's
- RRT's

# Generalized Voronoi Graph (GVG)



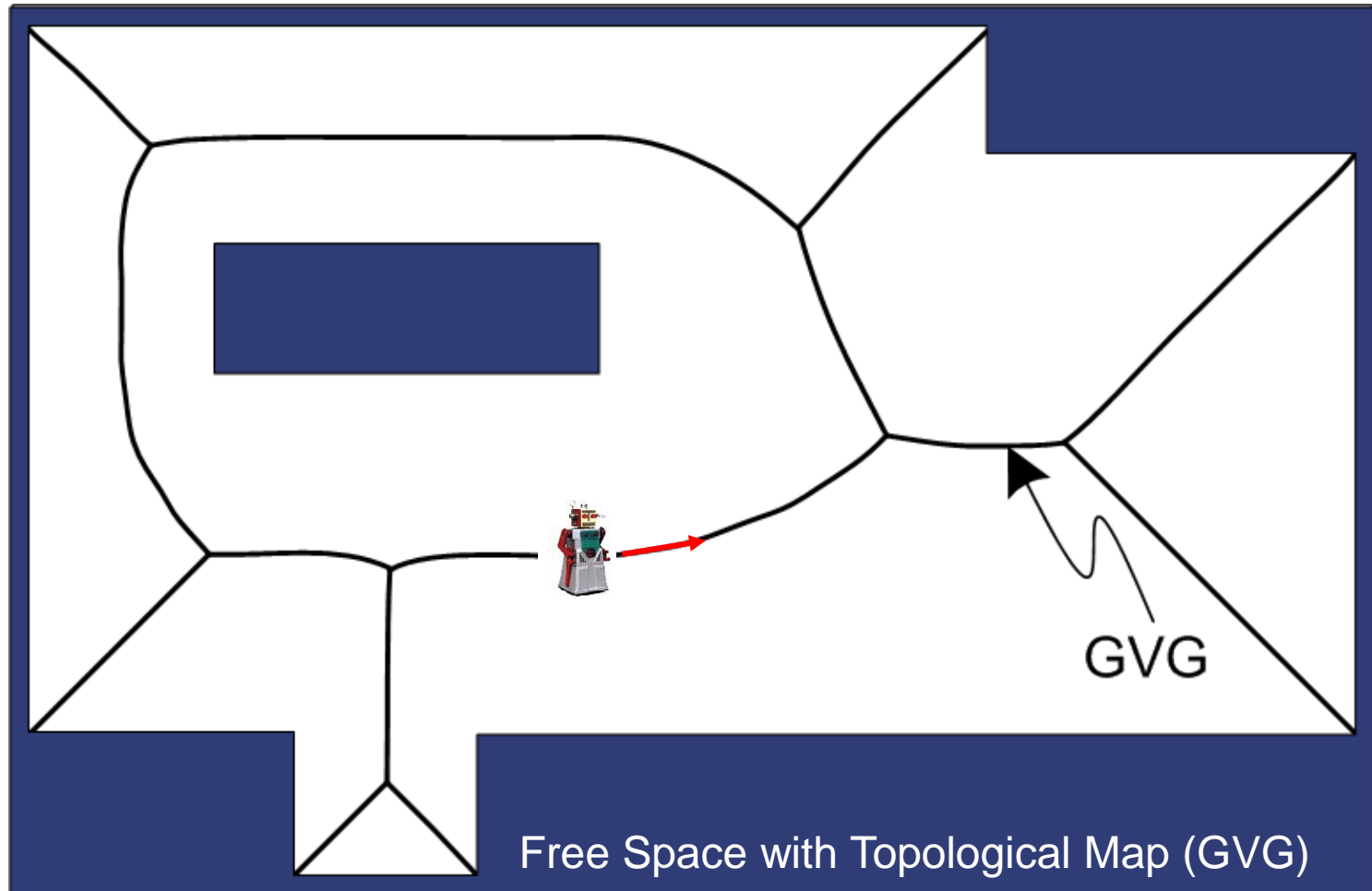
# Generalized Voronoi Graph (GVG)

- Access GVG



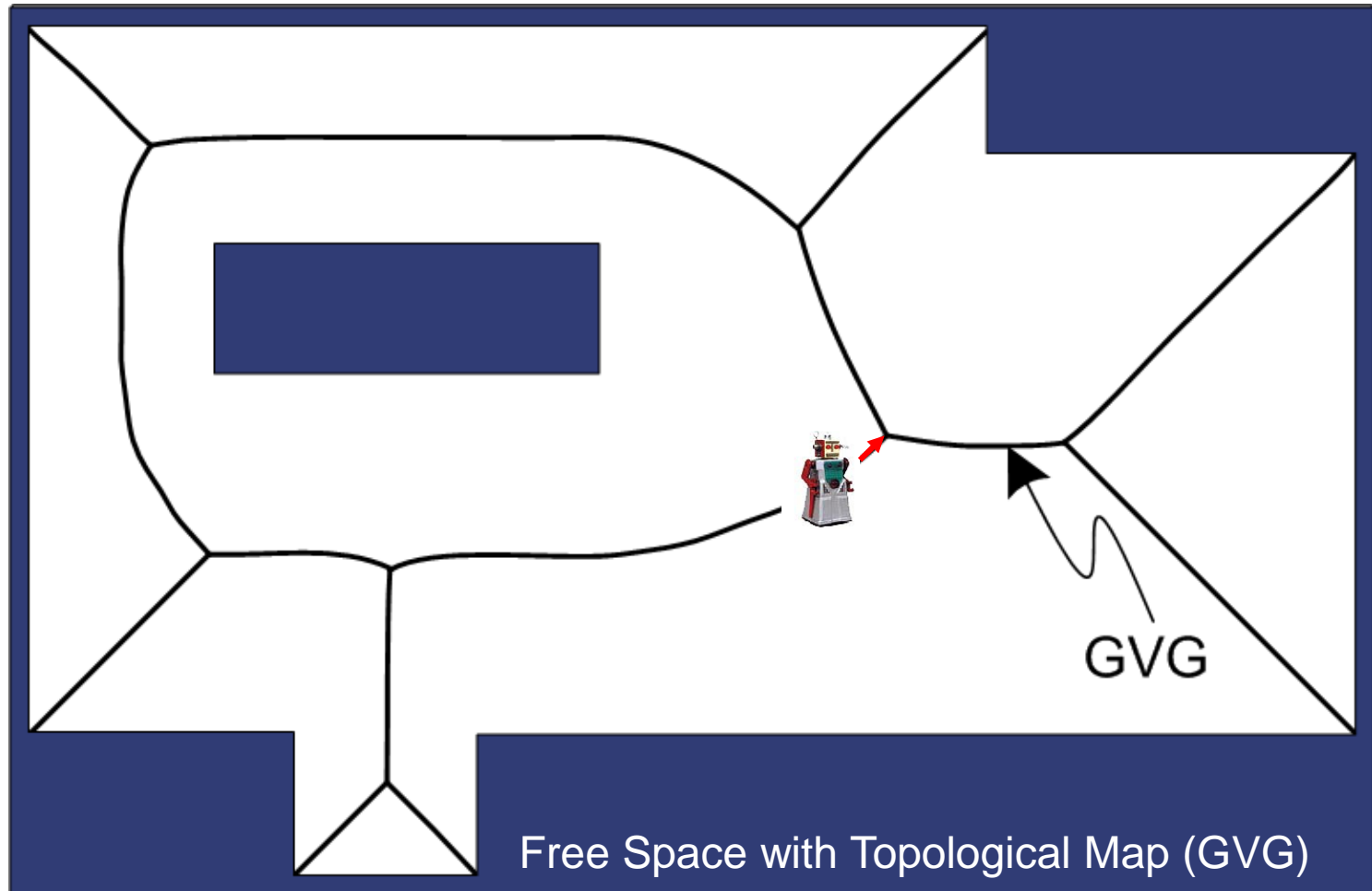
# Generalized Voronoi Graph (GVG)

- Access GVG
- Follow Edge



# Generalized Voronoi Graph (GVG)

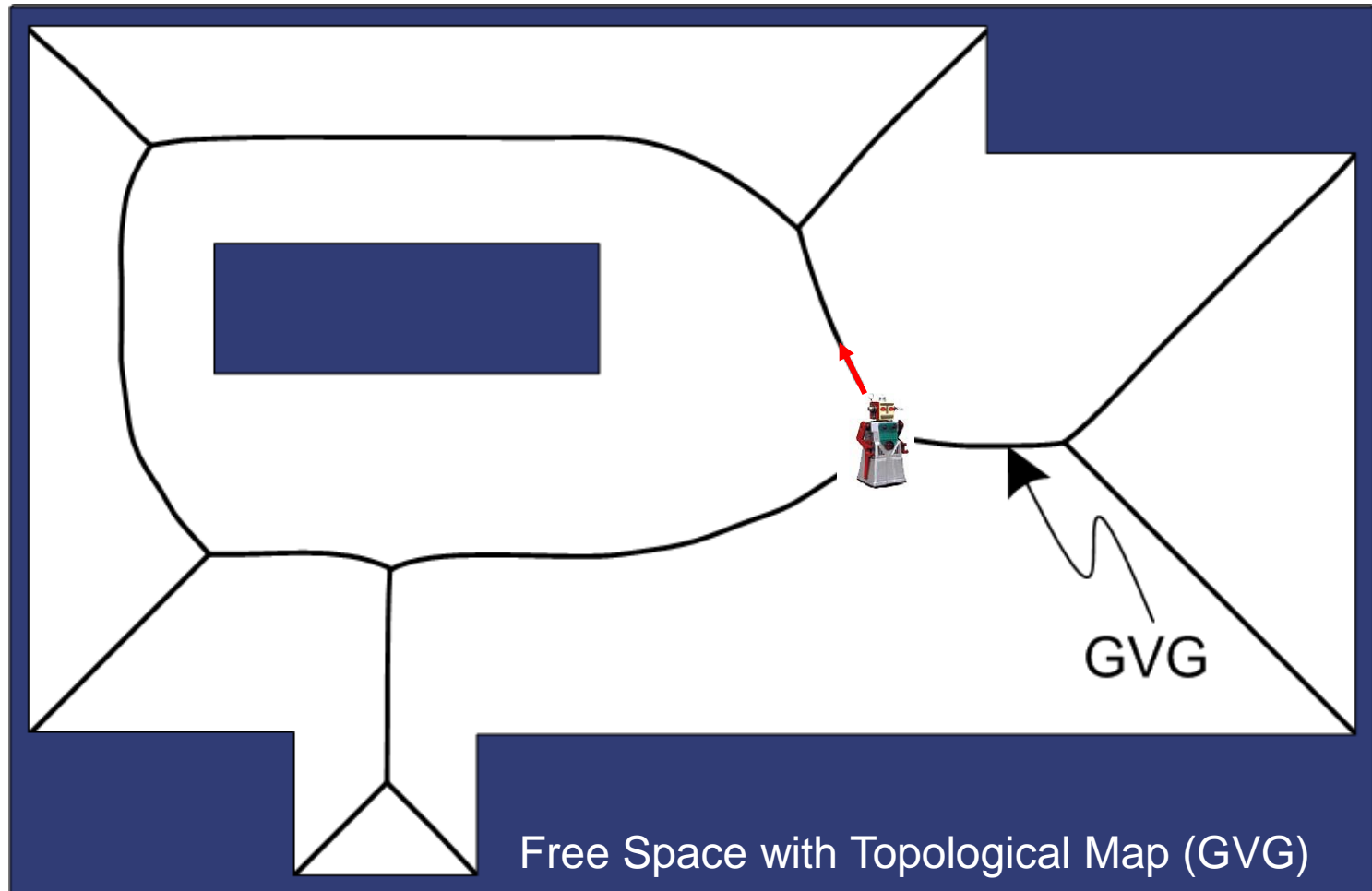
- Access GVG
- Home to the MeetPoint
- Follow Edge





# Generalized Voronoi Graph (GVG)

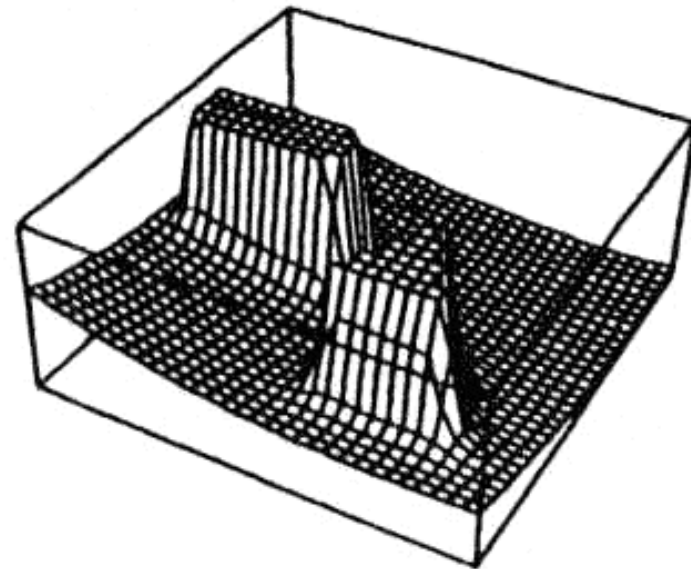
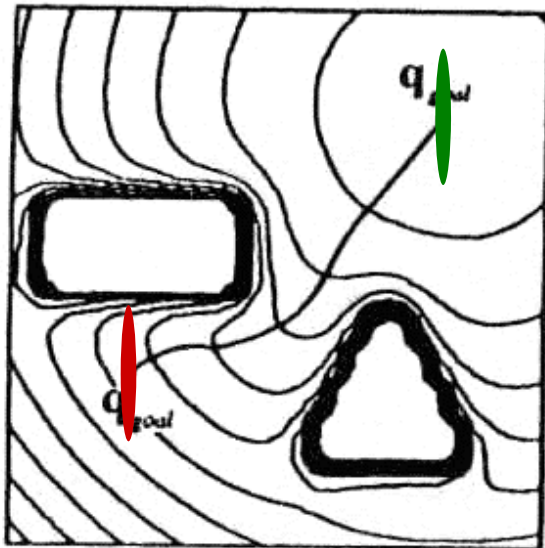
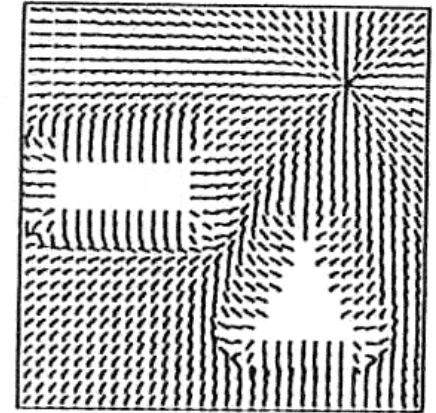
- Access GVG
- Home to the MeetPoint
- Follow Edge
- Select Edge



# Local techniques

## Potential Field methods

- compute a repulsive force away from obstacles
  - compute an attractive force toward the goal
- let the sum of the forces control the robot



To a large extent, this is computable from sensor readings

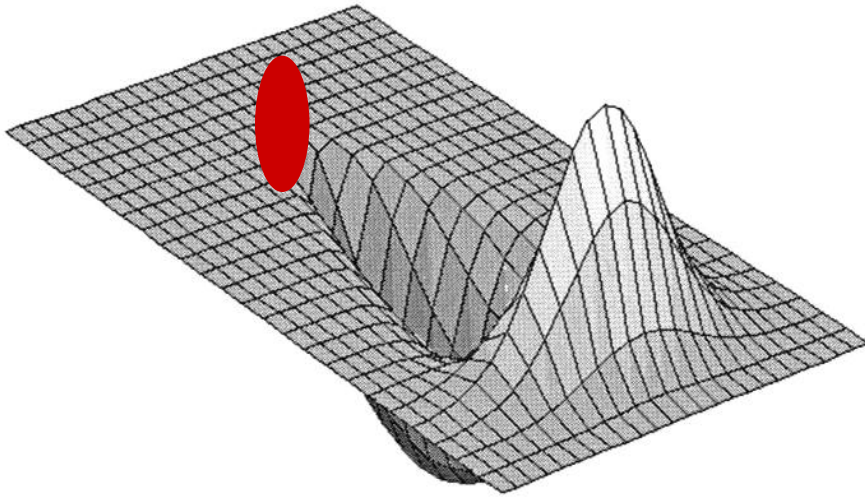
# SONAR modeling using Occupancy Grids

---

---

- The key to making accurate maps is combining lots of data.
- But combining these numbers means we have to know what they are !

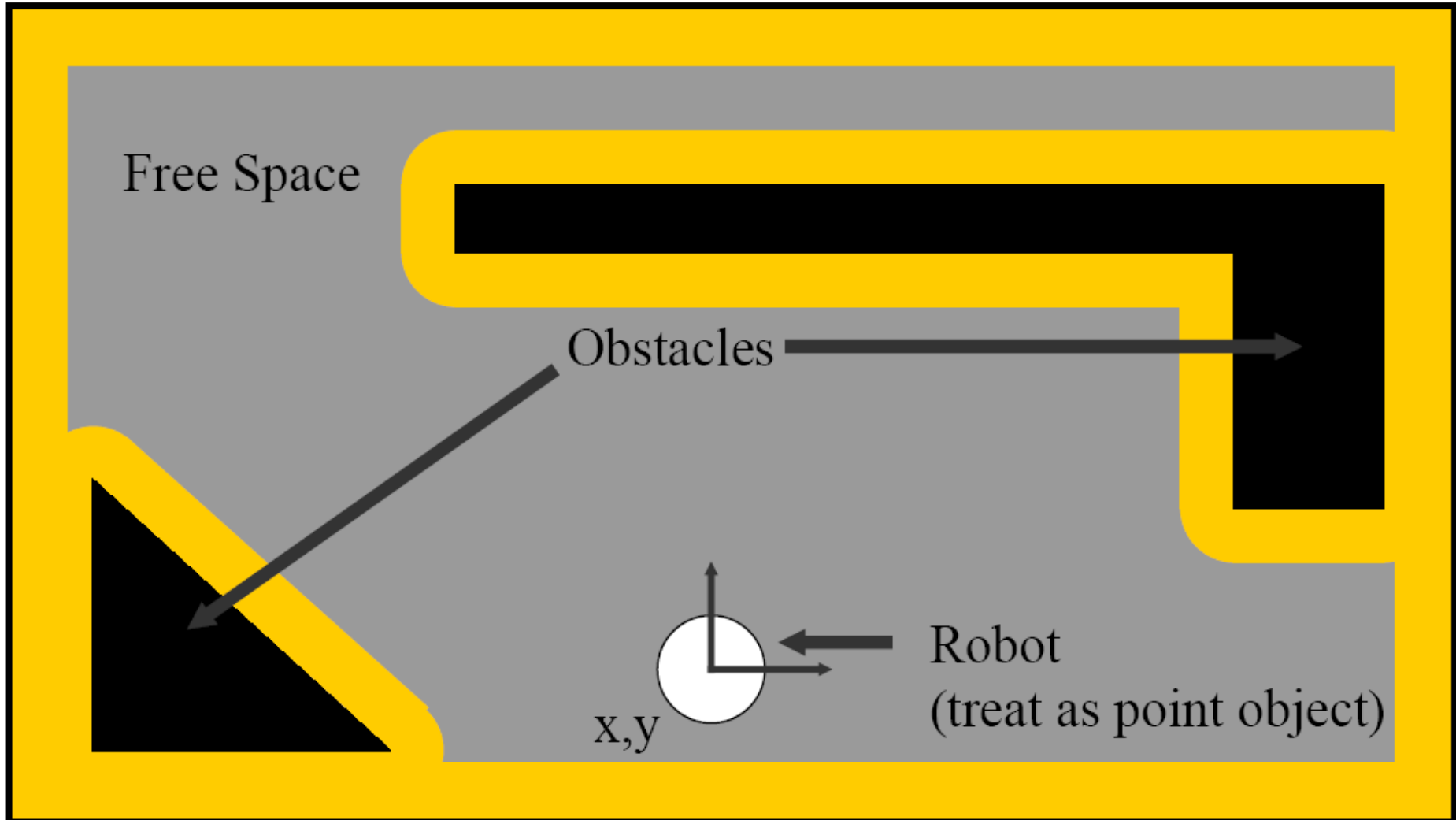
What should our map contain ?



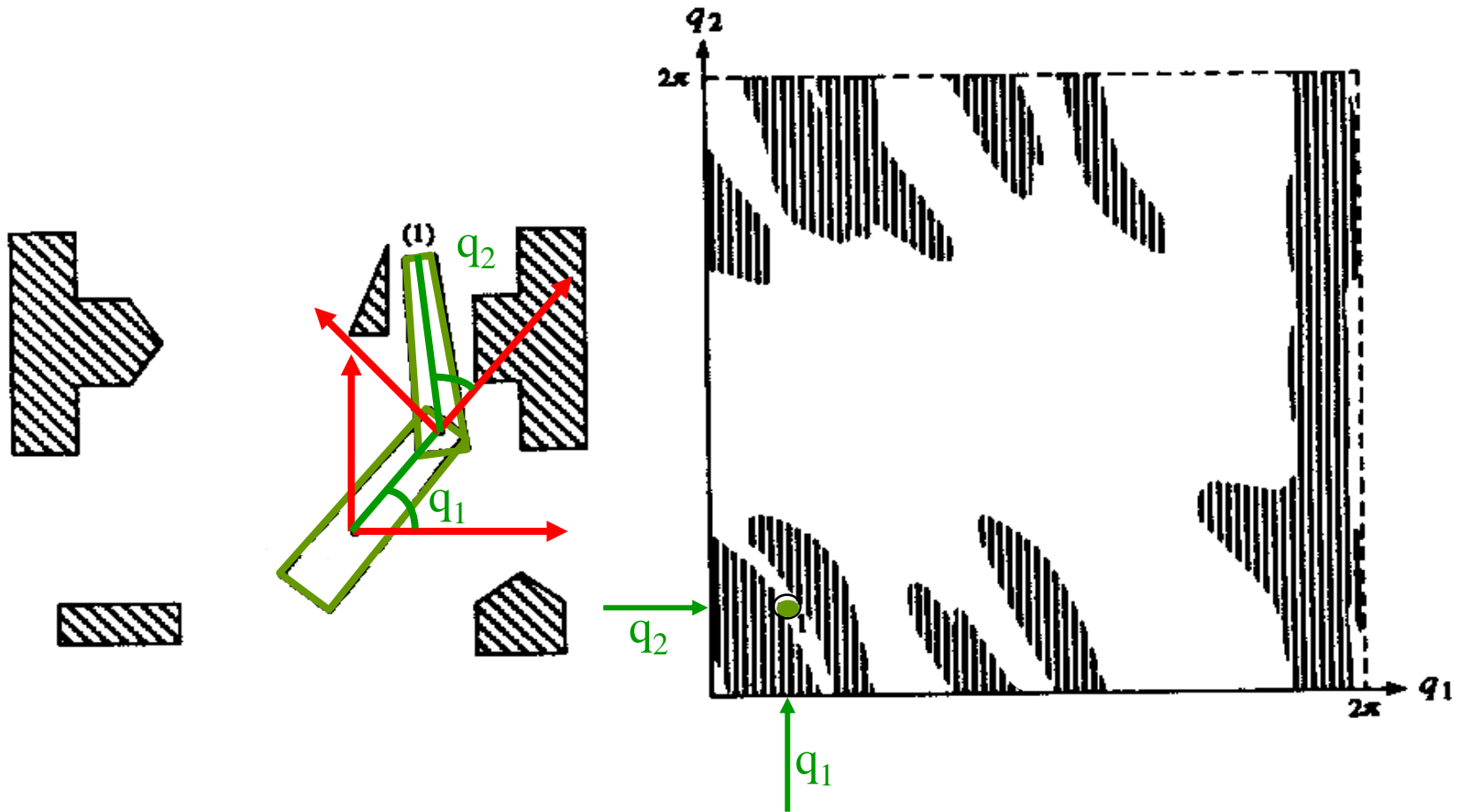
- small cells
- each represents a bit of the robot's environment
- larger values => obstacle
- smaller values => free

what is in each cell of this sonar model / map ?

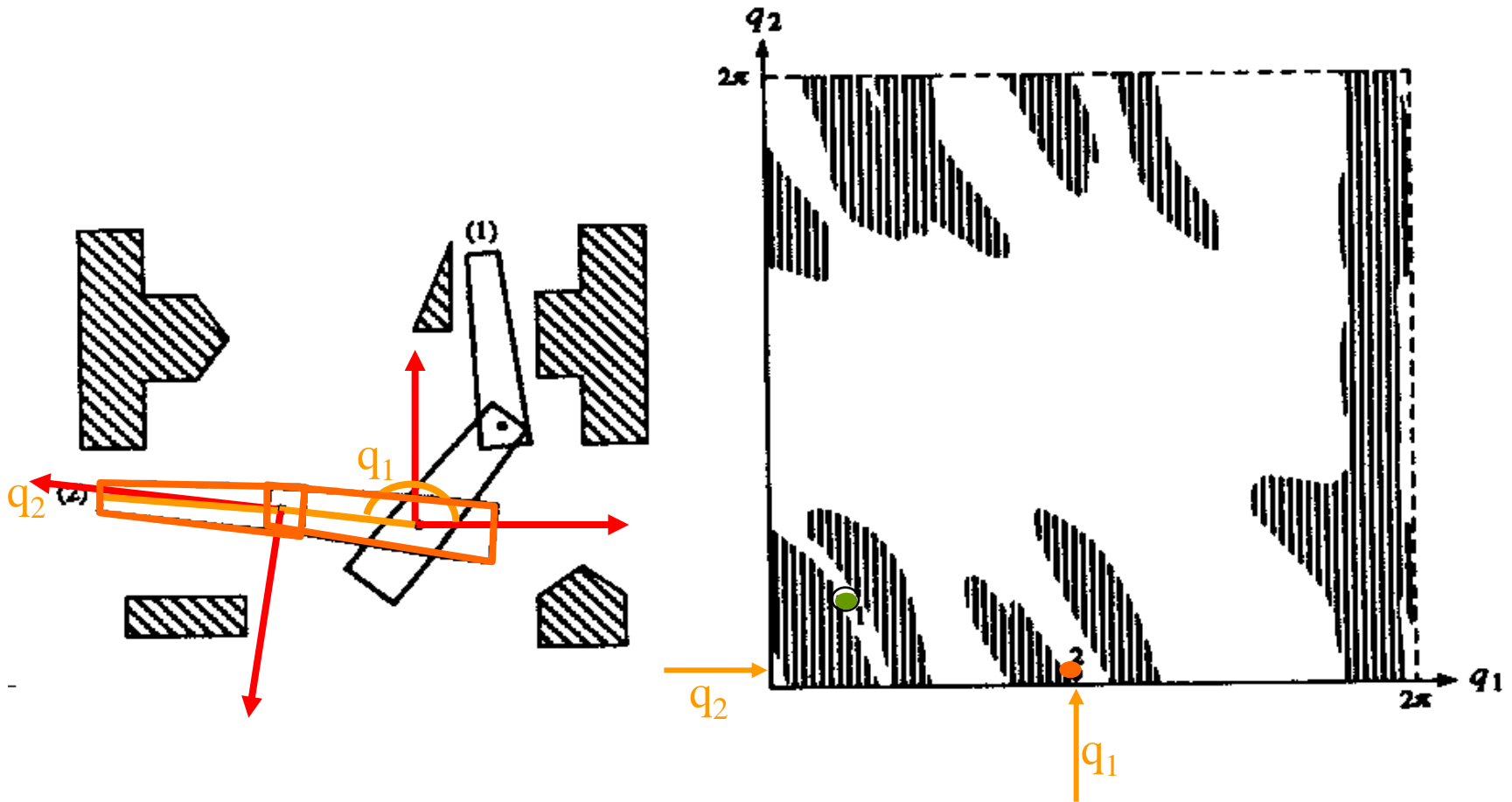
# Configuration Space



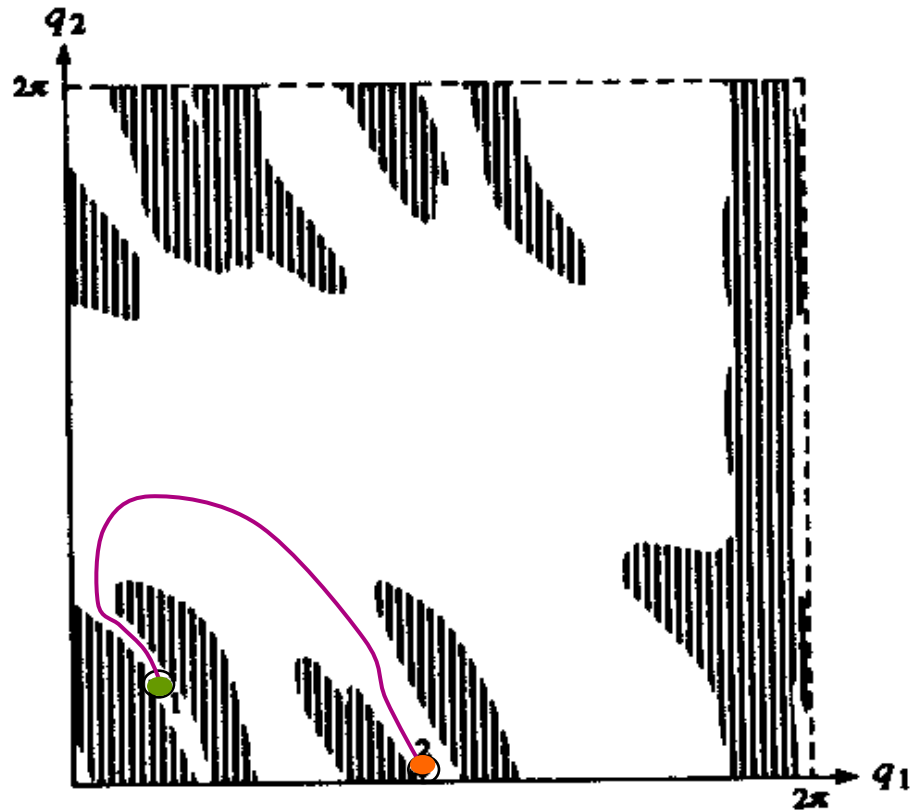
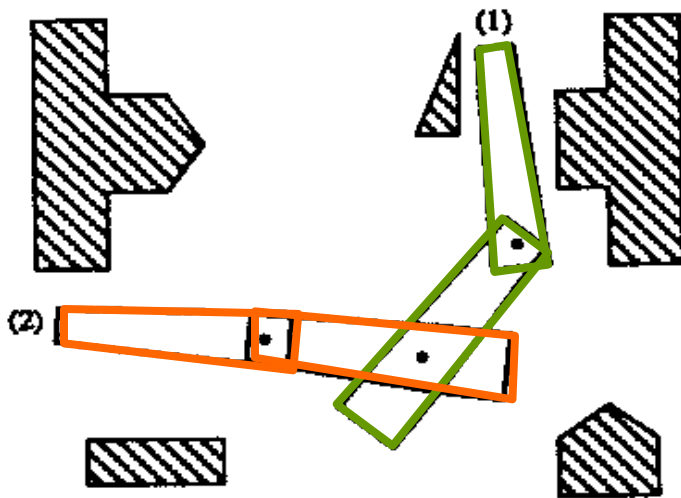
# Tool: Configuration Space (C-Space $C$ )



# Tool: Configuration Space (C-Space $C$ )



# Tool: Configuration Space (C-Space $C$ )

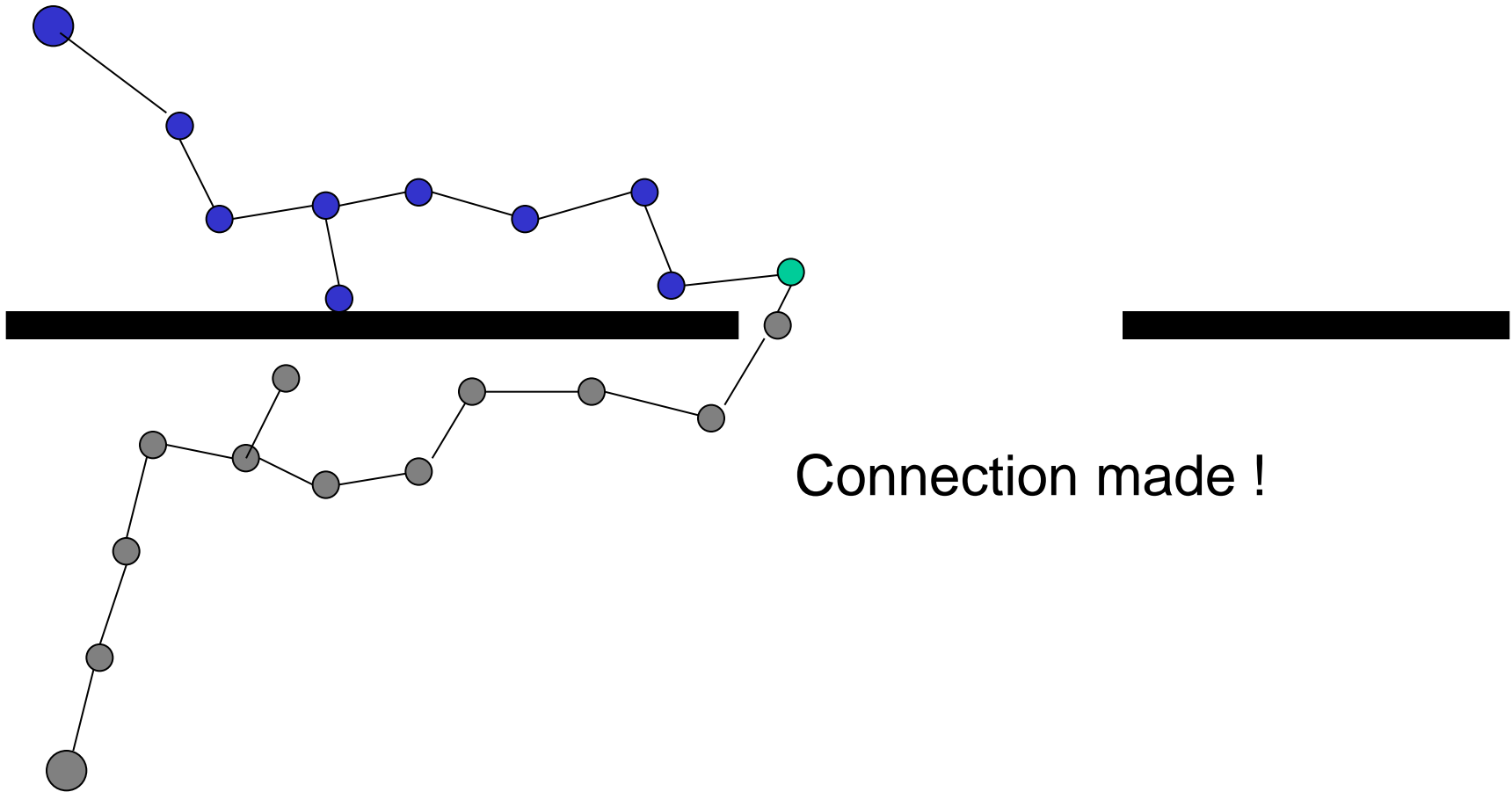


# Road Maps

- PRMs
- RRTs



# RRT-Connect: example



# Basic Controller Functions

---

Proportional control:  $u(t) = K_p e(t)$   $\frac{U(s)}{E(s)} = K_p$

Integral control:  $u(t) = K_i \int_0^t e(t) dt$   $\frac{U(s)}{E(s)} = \frac{K_i}{s}$

Differential control:  $u(t) = K_d \frac{d}{dt} e(t)$   $\frac{U(s)}{E(s)} = K_d s$



# Effect of Controller Functions

---

- Proportional Action
  - Simplest Controller Function
- Integral Action
  - Eliminates steady-state error
  - Can cause oscillations
- Derivative Action (“rate control”)
  - Effective in transient periods
  - Provides faster response (higher sensitivity)
  - Never used alone

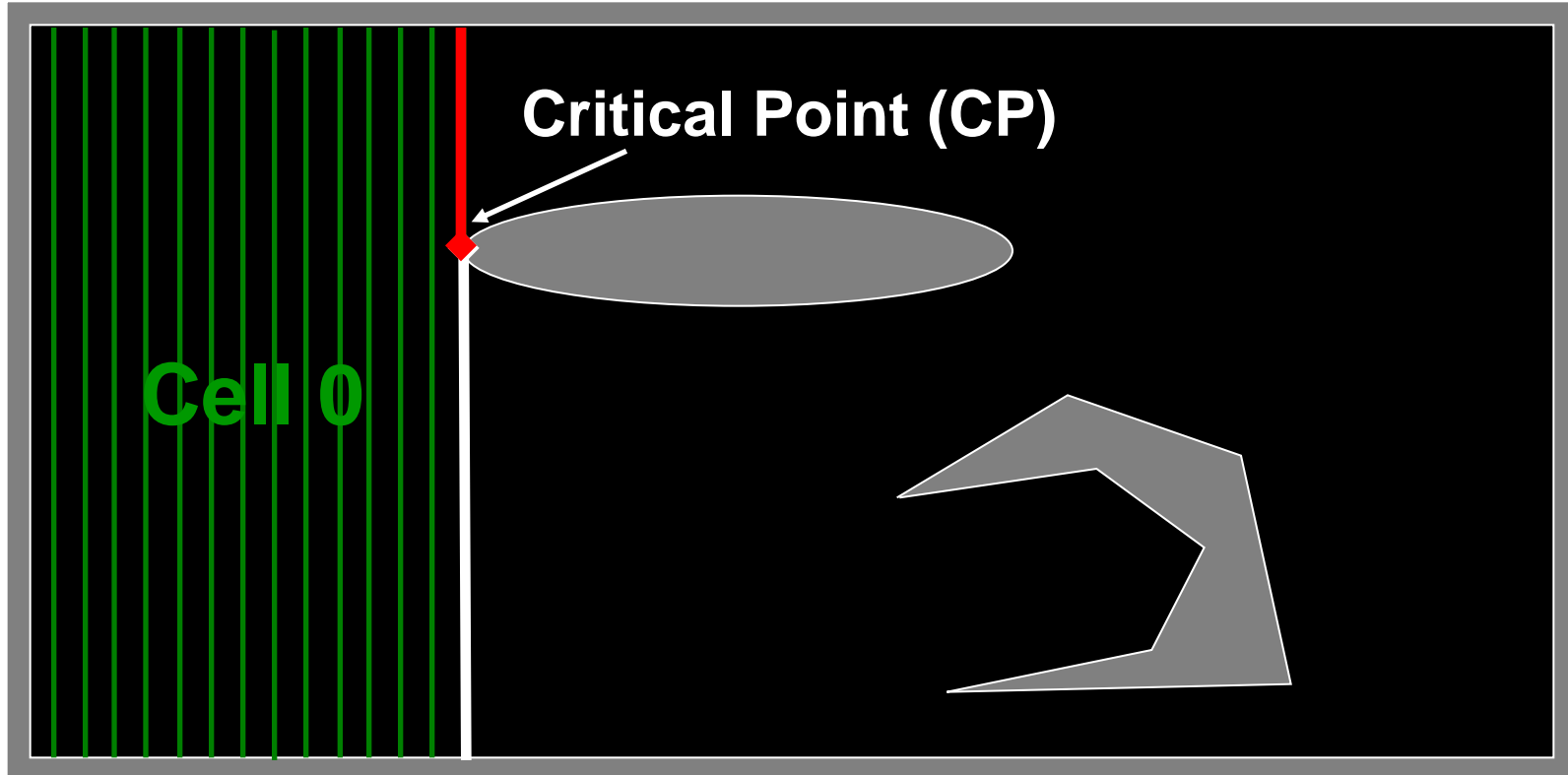




# Coverage

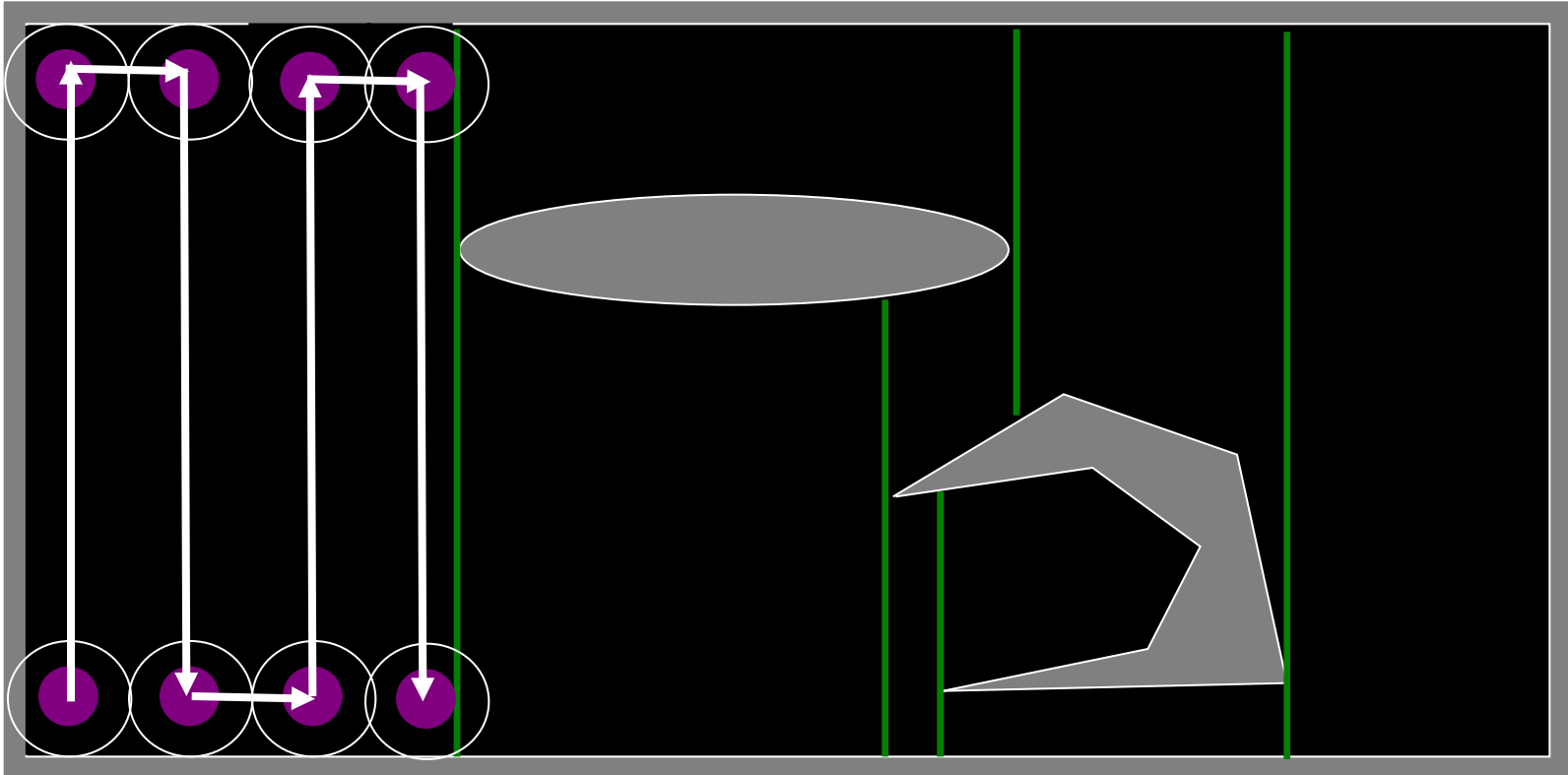
- First Distinction
  - Deterministic **Demining**
  - Random **Vacuum Cleaning**
- Second Distinction
  - Complete
  - No Guarantee
- Third Distinction
  - Known Environment
  - Unknown Environment

# Cellular Decomposition



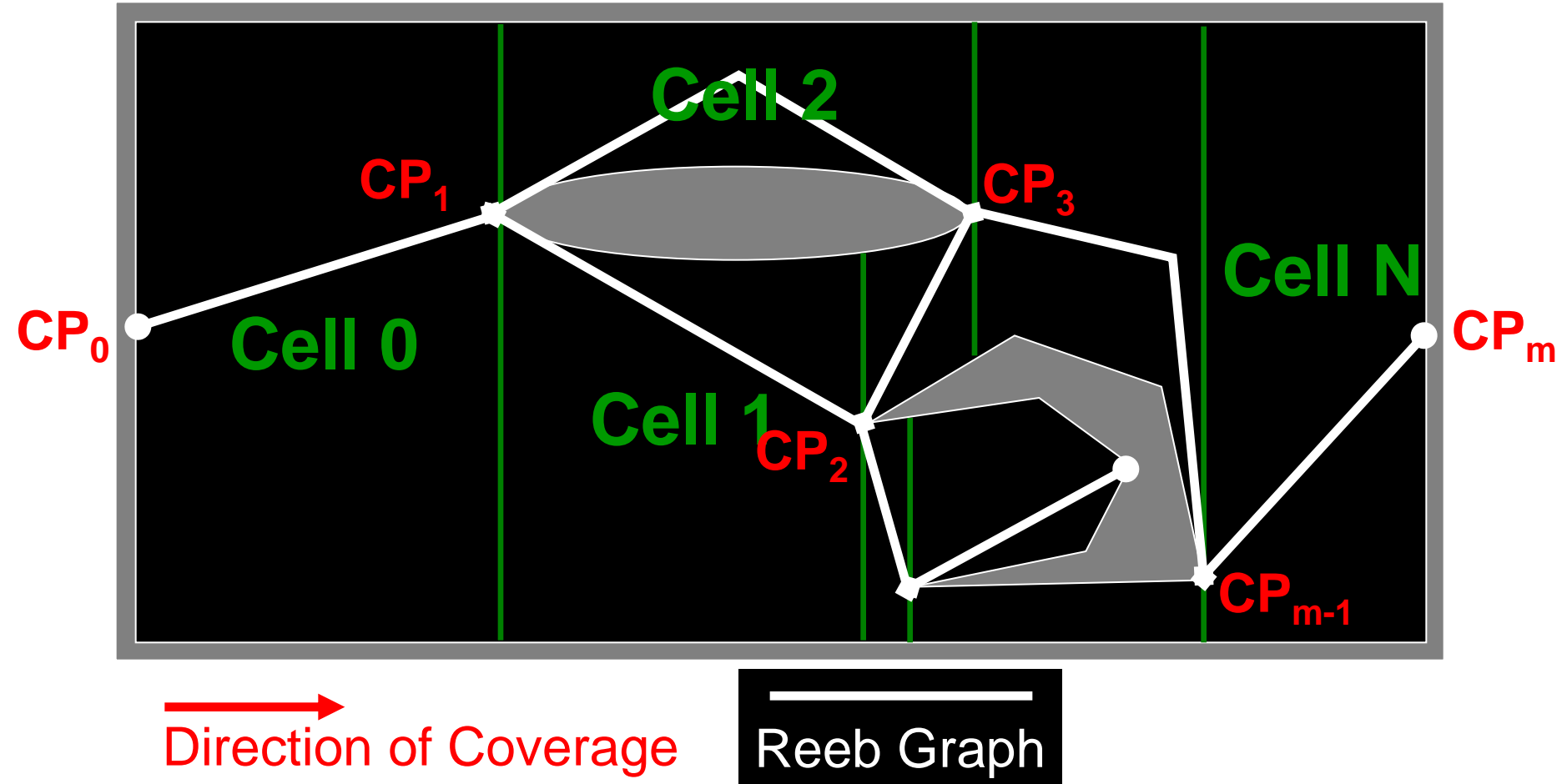
 Direction of Coverage

# Single Cell Coverage



Direction of Coverage

# Cellular Decomposition



# Multi-Robot Coverage

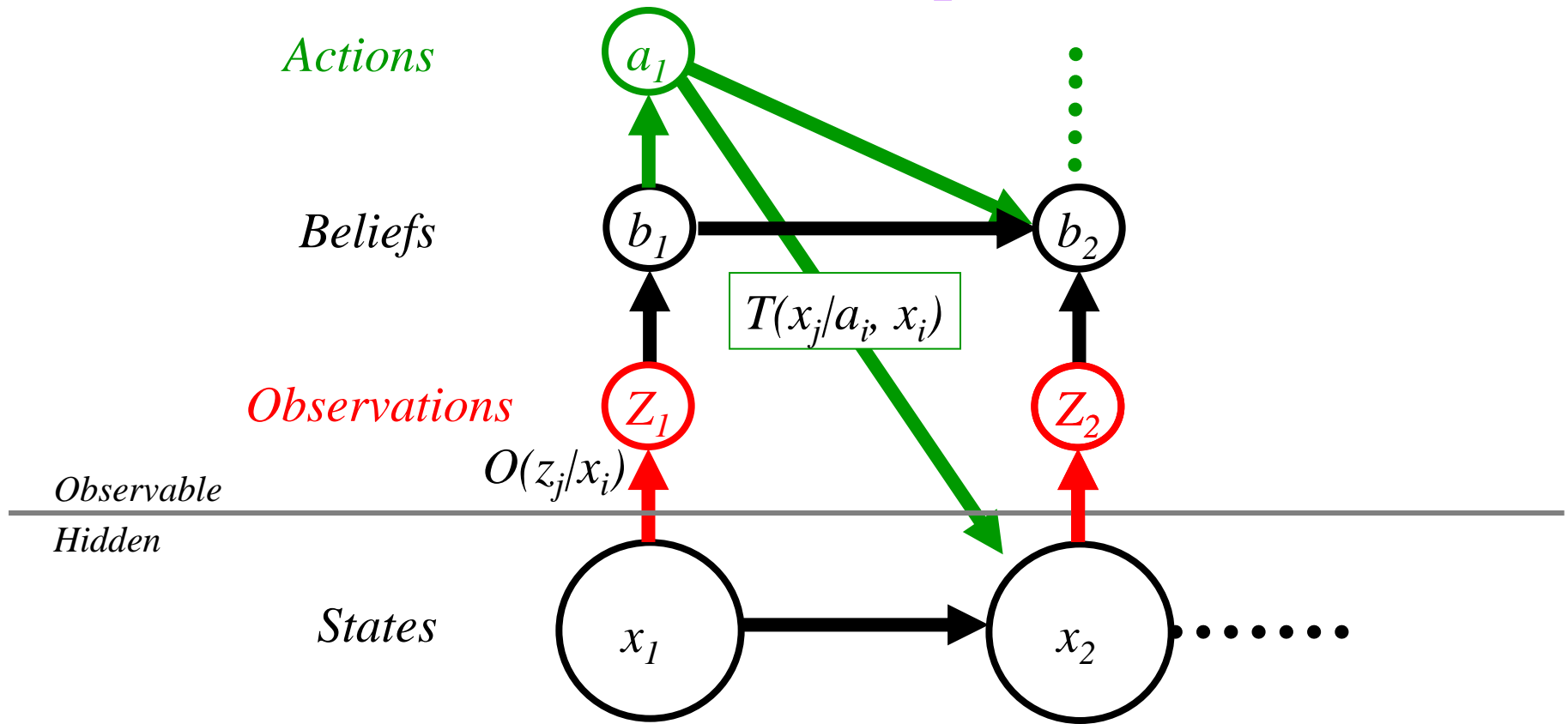
- Team based
- Distributed
  - Auctions



# Localization

- Tracking: Known initial position
- Global Localization: Unknown initial position
- Re-Localization: Incorrect known position
  - (kidnapped robot problem)

# Graphical Models, Bayes' Rule and the Markov Assumption



Bayes rule: 
$$p(x | y) = \frac{p(y | x) p(x)}{p(y)}$$

Markov : 
$$p(x_t | x_{t-1}, a_t, a_0, z_0, a_1, z_1, \dots, z_{t-1}) = p(x_t | x_{t-1}, a_t)$$

# Derivation of the Bayesian Filter

First-order Markov assumption shortens middle term:

$$Bel(x_t) = \eta \boxed{p(o_t | x_t)} \int p(x_t | x_{t-1}, a_{t-1}) p(x_{t-1} | a_{t-1}, \dots, o_0) dx_{t-1}$$

Finally, substituting the definition of  $Bel(x_{t-1})$ :

$$Bel(x_t) = \eta p(o_t | x_t) \int p(x_t | x_{t-1}, a_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

The above is the probability distribution that must be estimated from the robot's data

# Iterating the Bayesian Filter

- Propagate the motion model:

$$Bel_{-}(x_t) = \int P(x_t | a_{t-1}, x_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

Compute the current state estimate before taking a sensor reading by integrating over all possible previous state estimates and applying the motion model

- Update the sensor model:

$$Bel(x_t) = \eta P(o_t | x_t) Bel_{-}(x_t)$$

Compute the current state estimate by taking a sensor reading and multiplying by the current estimate based on the most recent motion history

# Different Approaches

## **Kalman filters** (late-60s?)

- Gaussians
- approximately linear models
- position tracking

## **Extended Kalman Filter**

## **Information Filter**

## **Unscented Kalman Filter**

## **Multi-hypothesis** ('00)

- Mixture of Gaussians
- Multiple Kalman filters
- Global localization, recovery

## **Discrete approaches** ('95)

- Topological representation ('95)
- uncertainty handling (POMDPs)
- occas. global localization, recovery
- Grid-based, metric representation ('96)
- global localization, recovery

## **Particle filters** ('98)

- Condensation (Isard and Blake '98)
- Sample-based representation
- Global localization, recovery
- Rao-Blackwellized Particle Filter

# The Kalman Filter

- Motion model is Gaussian...
- Sensor model is Gaussian...
- Each belief function is uniquely characterized by its mean  $\mu$  and covariance matrix  $\Sigma$
- Computing the posterior means computing a new mean  $\mu$  and covariance  $\Sigma$  from old data using actions and sensor readings
- *What are the key limitations?*
  - 1) Unimodal distribution
  - 2) Linear assumptions

# What we know...

# What we don't know...

- We know what the control inputs of our process are
  - We know what we've told the system to do and have a model for what the expected output should be if everything works right
- We don't know what the noise in the system truly is
  - We can only estimate what the noise might be and try to put some sort of upper bound on it
- When estimating the state of a system, we try to find a set of values that comes as close to the truth as possible
  - There will always be some mismatch between our estimate of the system and the true state of the system itself. We just try to figure out how much mismatch there is and try to get the best estimate possible

# Kalman Filter Components

(also known as: Way Too Many Variables...)

Linear discrete time dynamic system (motion model)

$$x_{t+1} = F_t x_t + B_t u_t + G_t w_t$$

State transition function      Control input function      Noise input function with covariance Q

The diagram shows the state transition equation  $x_{t+1} = F_t x_t + B_t u_t + G_t w_t$ . Arrows point from the following labels to the corresponding terms in the equation: 'State' points to  $x_t$ ; 'Control input' points to  $u_t$ ; 'Process noise' points to  $w_t$ ; 'State transition function' points to  $F_t$ ; 'Control input function' points to  $B_t$ ; and 'Noise input function with covariance Q' points to  $G_t$ .

Measurement equation (sensor model)

$$z_{t+1} = H_{t+1} x_{t+1} + n_{t+1}$$

Sensor reading      State      Sensor noise with covariance R

Sensor function

The diagram shows the measurement equation  $z_{t+1} = H_{t+1} x_{t+1} + n_{t+1}$ . Arrows point from the following labels to the corresponding terms in the equation: 'Sensor reading' points to  $z_{t+1}$ ; 'State' points to  $x_{t+1}$ ; 'Sensor noise with covariance R' points to  $n_{t+1}$ ; and 'Sensor function' points to  $H_{t+1}$ .

*Note: Write these down!!!*



# Computing the MMSE Estimate of the State and Covariance

What is the **minimum mean square error** estimate of the system state and covariance?

$$\hat{x}_{t+1|t} = F_t \hat{x}_{t|t} + B_t u_t \quad \text{Estimate of the state variables}$$

$$\hat{z}_{t+1|t} = H_{t+1} \hat{x}_{t+1|t} \quad \text{Estimate of the sensor reading}$$

$$P_{t+1|t} = F_t P_{t|t} F_t^T + G_t Q_t G_t^T \quad \text{Covariance matrix for the state}$$

$$S_{t+1|t} = H_{t+1} P_{t+1|t} H_{t+1}^T + R_{t+1} \quad \text{Covariance matrix for the sensors}$$

# The Kalman Filter...

Propagation (motion model):

$$\hat{x}_{t+1/t} = F_t \hat{x}_{t/t} + B_t u_t$$

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

Update (sensor model):

$$\hat{z}_{t+1} = H_{t+1} \hat{x}_{t+1/t}$$

$$r_{t+1} = z_{t+1} - \hat{z}_{t+1}$$

$$S_{t+1} = H_{t+1} P_{t+1/t} H_{t+1}^T + R_{t+1}$$

$$K_{t+1} = P_{t+1/t} H_{t+1}^T S_{t+1}^{-1}$$

$$\hat{x}_{t+1/t+1} = \hat{x}_{t+1/t} + K_{t+1} r_{t+1}$$

$$P_{t+1/t+1} = P_{t+1/t} - P_{t+1/t} H_{t+1}^T S_{t+1}^{-1} H_{t+1} P_{t+1/t}$$

# ...but what does that mean in English???

Propagation (motion model):

$$\hat{x}_{t+1/t} = F_t \hat{x}_{t/t} + B_t u_t$$

- State estimate is updated from system dynamics

$$P_{t+1/t} = F_t P_{t/t} F_t^T + G_t Q_t G_t^T$$

- Uncertainty estimate *GROWS*

Update (sensor model):

$$\hat{z}_{t+1} = H_{t+1} \hat{x}_{t+1/t}$$

- Compute expected value of sensor reading

$$r_{t+1} = z_{t+1} - \hat{z}_{t+1}$$

- Compute the difference between expected and “true”

$$S_{t+1} = H_{t+1} P_{t+1/t} H_{t+1}^T + R_{t+1}$$

- Compute covariance of sensor reading

$$K_{t+1} = P_{t+1/t} H_{t+1}^T S_{t+1}^{-1}$$

- Compute the Kalman Gain (how much to correct est.)

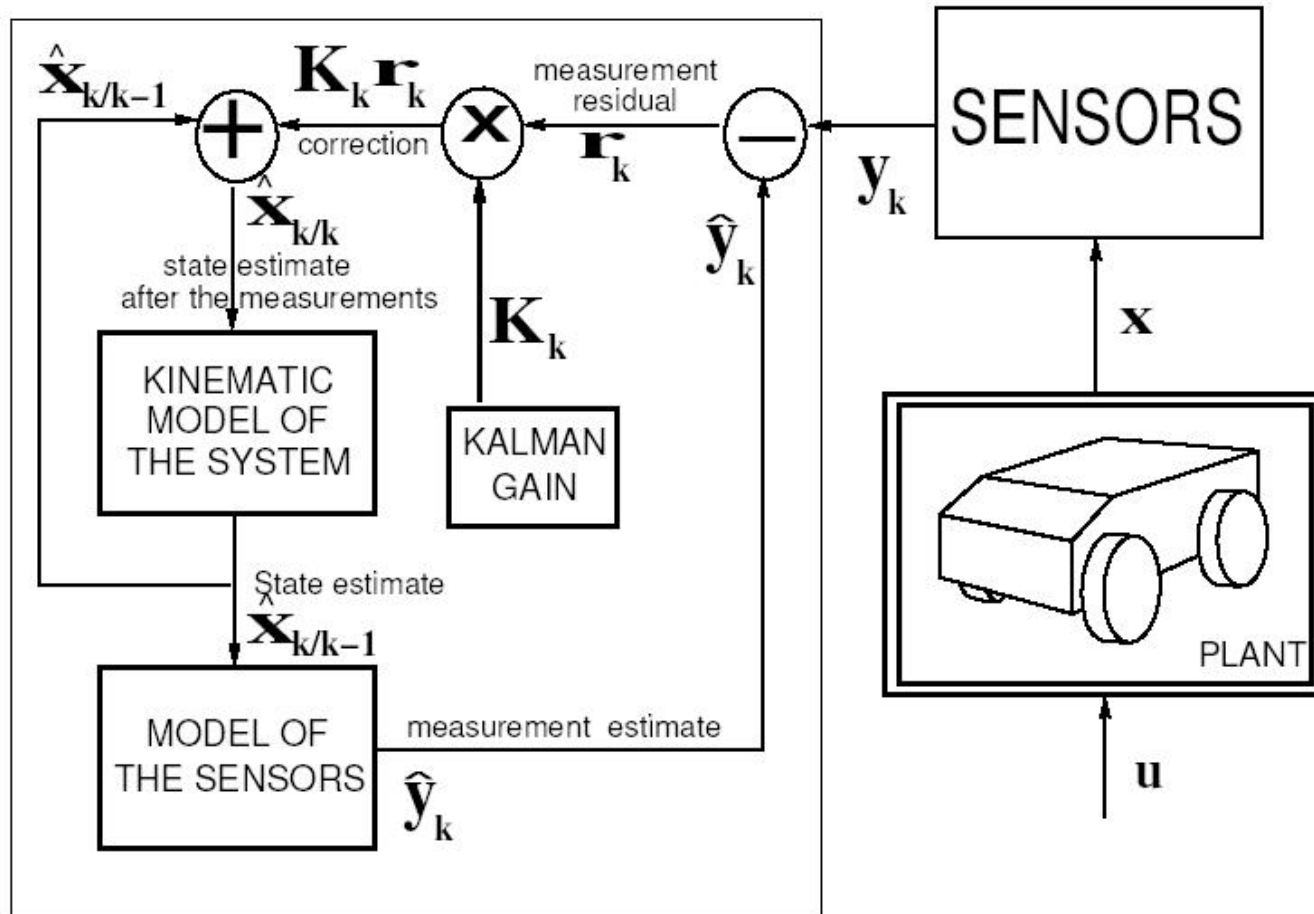
$$\hat{x}_{t+1/t+1} = \hat{x}_{t+1/t} + K_{t+1} r_{t+1}$$

- Multiply residual times gain to correct state estimate

$$P_{t+1/t+1} = P_{t+1/t} - P_{t+1/t} H_{t+1}^T S_{t+1}^{-1} H_{t+1} P_{t+1/t}$$

- Uncertainty estimate *SHRINKS*

# Kalman Filter Block Diagram



# Calculation of $\tilde{\phi}_{t+1}$

$$\begin{aligned}\tilde{\phi}_{t+1} &= \phi_{t+1} - \hat{\phi}_{t+1} \\ &= \phi_t + \omega_t \Delta t - \hat{\phi}_t - (\omega_t + w_\omega) \Delta t \\ &= \tilde{\phi}_t - w_\omega \Delta t\end{aligned}$$

# Calculation of

$$\tilde{x}_{t+1} = x_{t+1} - \hat{x}_{t+1}$$

$$\tilde{y}_{t+1} = y_{t+1} - \hat{y}_{t+1}$$

$$\tilde{x}_{t+1} = x_{t+1} - \hat{x}_{t+1}$$

$$\tilde{x}_{t+1} = x_{t+1} - \hat{x}_{t+1}$$

$$= x_t + v_t \Delta t \cos(\phi_t) - \hat{x}_t - (v_t - w_v) \Delta t \cos(\hat{\phi}_t)$$

$$= x_t - \hat{x}_t + v_t \Delta t \cos(\phi_t) - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t)$$

$$= \tilde{x}_t + v_t \Delta t \cos(\tilde{\phi}_t + \hat{\phi}_t) - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t)$$

$$= \tilde{x}_t + v_t \Delta t [\cos(\tilde{\phi}_t) \cos(\hat{\phi}_t) + \sin(\tilde{\phi}_t) \sin(\hat{\phi}_t)] - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t)$$

$$\cong \tilde{x}_t + v_t \Delta t \cos(\hat{\phi}_t) - v_t \Delta t \tilde{\phi}_t \sin(\hat{\phi}_t) - v_t \Delta t \cos(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t)$$

$$= \tilde{x}_t - v_t \Delta t \tilde{\phi}_t \sin(\hat{\phi}_t) + w_v \Delta t \cos(\hat{\phi}_t)$$

$$\tilde{y}_{t+1} = y_{t+1} - \hat{y}_{t+1}$$

= ...

# Covariance Estimation

$$\begin{aligned} P_{t+1/t} &= E[\tilde{X}_{t+1} \tilde{X}_{t+1}^T] \\ &= E[(F_t \tilde{X}_t + G_t w_t)(F_t \tilde{X}_t + G_t w_t)^T] \\ &= F_t E[\tilde{X}_t \tilde{X}_t^T] F_t^T + G_t E[w_t w_t^T] G_t^T \\ &= F_t P_{t/t} F_t^T + G_t Q_t G_t^T \end{aligned}$$

where

$$Q_t = E[w_t w_t^T] = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\omega^2 \end{bmatrix}$$

# Some observations

- The larger the error, the smaller the effect on the final state estimate
  - If *process* uncertainty is larger, *sensor* updates will dominate state estimate
  - If *sensor* uncertainty is larger, *process* propagation will dominate state estimate
- Improper estimates of the state and/or sensor covariance may result in a rapidly diverging estimator
  - As a rule of thumb, the residuals must always be bounded within a  $\pm 3\sigma$  region of uncertainty
  - This measures the “health” of the filter
- *Many* propagation cycles can happen between updates



# Heuristic Search Planning Method

---

- Solution to exploration planning for camera sensor networks
  - Composed of two alternated steps: exploration and re-localization
  - Combined distance and uncertainty cost function
  - Heuristic search for good paths



# Exploration and Uncertainty Reduction

---

- Decision (exploration vs exploitation)
- Target Node
- Path Planning through the known graph
- Exploration Strategies



# Exploration and Uncertainty Reduction

---

- Decision (exploration vs. exploitation)
  - **Epsilon-Greedy**
  - **Epsilon-First**
  - **Adaptive**
  - **Bounded Uncertainty**
- Target Node
- Path Planning through the known graph
- Exploration Strategies



# Exploration and Uncertainty Reduction

---

- Decision (exploration vs. exploitation)
- Target Node (Exploration)
  - **Random**
  - **Shortest distance**
  - **Maximum Uncertainty**
  - **Minimum Uncertainty**
- Path Planning through the known graph
- Exploration Strategies



# Exploration and Uncertainty Reduction

---

- Decision (exploration vs. exploitation)
- Target Node (Relocalization)
  - **Maximum Uncertainty**
- Path Planning through the known graph
- Exploration Strategies



# Exploration and Uncertainty Reduction

- Decision (exploration vs. exploitation)
- Target Node
- Path Planning through the known graph
  - Work with D. Meger and G. Dudek [IROS 2008]
  - A\* based strategy
  - Cost:  $C(p) = \omega_d \text{length}(p) + \omega_u \text{trace}(P(p))$
  - Distance-based “cost-to-go” heuristic function  $h$  used to compute estimated cost

$$C(n) = f(n) + h(n)$$

Estimated cost through n

Cost so far

Estimated cost to go

- Exploration Strategies



# Exploration and Uncertainty Reduction

---

- Decision (exploration vs. exploitation)
- Target Node
- Path Planning through the known graph
- **Exploration Strategies**
  - One Step Exploration
  - Ear based exploration (submitted to IROS 2012)



# Computer Vision

- Projection 3D->2D
- Correspondence Problem
- Stereo
- Optical Flow
- Features