Introduction to Multi-Agent Programming

6. Cooperative Sensing

Modeling Sensors, Kalman Filter,
Markov Localization, Potential Fields

Alexander Kleiner, Bernhard Nebel

Contents

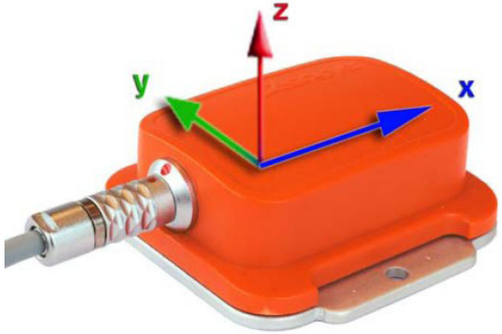
Introduction

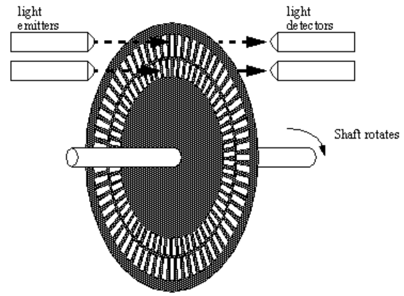
- Cooperative sensing and world modeling follows the goal of deliberative decision making
  - Which is in contrast to reactive acting
- Agents perceive their environment by sensors
  - However, sensing can either be inaccurate or ambiguous
  - Sensing requires the process of world modeling for meaningful and robust decision making
  - Probabilistic models are the first choice for this task
- World models can be used to extract abstract predicates of the world
  - For example, "***objectInOpponentGoal****(ball)*"

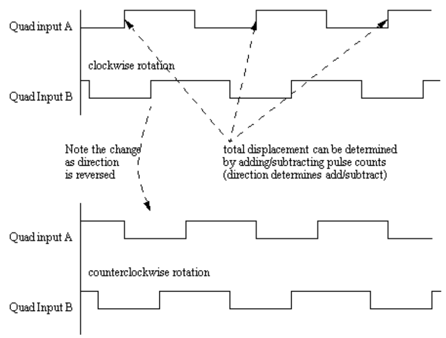
Processing Sensor data I

Inertial Measurement Unit (IMU) & Wheel Odometry

- A closed system for detecting orientation and motion of a vehicle or human
- Typically consists of 3 accelerometer, 3 gyroscopes, and 3 magnetometers
- Data rate @100 Hz
- Gyro reliable only within some time period (temperature drift)
- Magnetometer data can locally be wrong (magnetic perturbation)
- Therefore, gyro, accelerometer, and magnetometer data is fused by a Kalman Filter onboard the IMU sensor
- For the estimation of robot poses (x,y,θ) also wheel odometry, a hardware that counts the number of wheel revolutions per second, is required







Processing Sensor data II

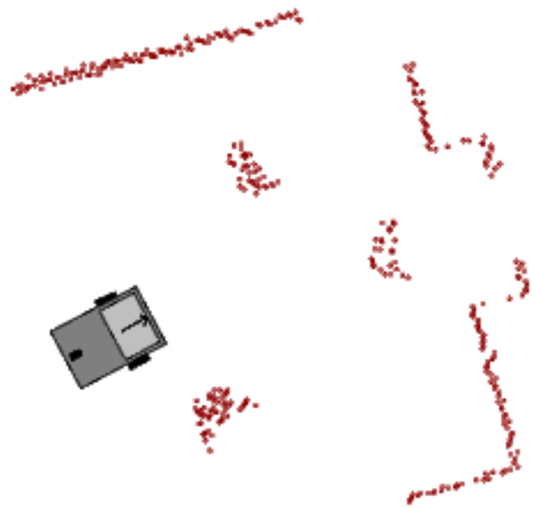
Laser Range Finders (LRFs)

- Found on many robots
- Highly accurate, high data rate
- Measures distances and angles to surrounding objects
- Returns distances d_i and angles α_i, with $i \in [0 \ldots FOV/resolution]$

SICK LMS200

Hokuyo URG

	Sick LMS200	Sick S300	Hokuyo URG-04LX
Weight	4500 g	1200 g	160 g
Volume	$\approx 20\ cm^3$	$\approx 15\ cm^3$	$\approx 5\ cm^3$
FOV	180°	270°	240°
Max. Range	80 m	30 m	4 m
Max. Ang. Res.	0.25°	0.5°	0.36°
Accuracy	± 15 mm	± 30 mm	± 10 mm
Scans per second	30	20	10
Interface	RS-232/RS-422	RS-232/RS-422	RS-232/USB



Scan taken on a soccer field

Processing Sensor data III

Color Cameras

- Sensor that generates color images, e.g. with 640x480 pixel resolution @30hz

Logitech Quickcam 4000 Pro

Sony DFW-500

- Can be used for object, e.g. ball detection
  - Color thresholding, e.g. separation of ball colors from background
  - Determination of relative object location by camera calibration or interpolation

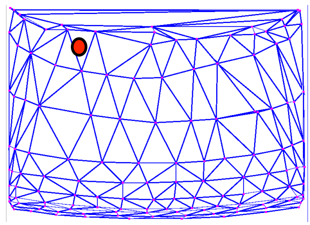
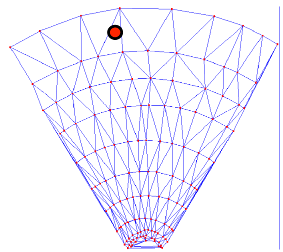


Image coordinates



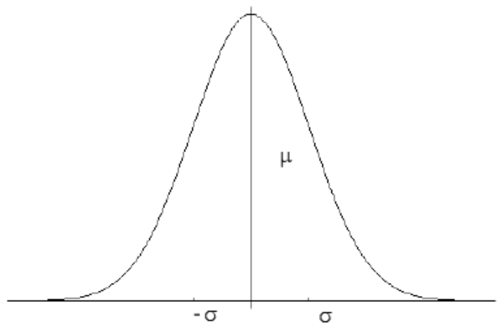
World coordinates

Modeling Sensor noise

- Sensor data is typically noisy, .e.g., the distance measurement of a LRF at one meter can be 1m ± 1cm
- Sensor noise is typically modeled by a normal distribution
- Fully described by mean μ and variance σ^2

one-dimensional:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} e^{-\frac{1}{2}\left(\frac{x-\mu_x}{\sigma_x}\right)^2}$$



Notation:

$$x \sim N(\mu_x, \sigma_x^2)$$

N-dimensional:

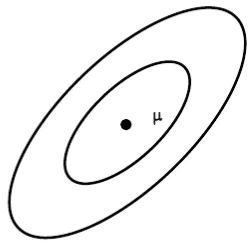
$$p(x) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma_x)}} e^{-\frac{1}{2}(x-\mu_x)^T \Sigma_x^{-1} (x-\mu_x)}$$

Notation:

$$x \sim N(\mu_x, \Sigma_x)$$

n-dimensional vector

n x n matrix



Transformation of density functions I

Linear Transformation

- When processing data from multiple sensors, all observations have to be transformed into a single coordinate system
- For example, distance and angle measurements of a LRF have to be integrated into a Cartesian coordinate frame
- Linear transformations can be represented by $F(u) = Au + b$ where *A* is a *nXm* Matrix and *b* a n-dimensional vector

- Given: μ_u Σ_u Wanted: new mean and variance μ_u Σ_u
- Mean μ_x and covariance Σ_x can be computed by:

$$\begin{aligned}\mu_x &= E(x)\\ &= E(Au+b)\\ &= AE(u)+b\\ &= A\mu_u + b\end{aligned}$$

$$\begin{aligned}\Sigma_x &= E((x-E(x))(x-E(x))^T)\\ &= E((Au+b-AE(u)-b)(Au+b-AE(u)-b)^T)\\ &= E((A(u-E(u)))(A(u-E(u)))^T)\\ &= E((A(u-E(u)))((u-E(u))^T A^T))\\ &= AE((u-E(u))(u-E(u))^T)A^T\\ &= A\Sigma_u A^T\end{aligned}$$

Transformation of density functions II

Non-Linear Transformation

- Linearization is necessary in order to yield a normal distribution
  - Approximation by Taylor polynom while skipping higher order terms:

  where $\nabla F(\hat{u}) = \frac{\partial F}{\partial u}(\hat{u})$ is a *n X m* Matrix (also known as Jacobi- Matrix) with partial derivates of *F* at $\hat{u}$.

- Notation according to linear case:

$$\begin{aligned} A &= \nabla F(\mu_u) \\ b &= F(\mu_u) - \nabla F(\mu_u)\mu_u \end{aligned}$$

- Mean μ_x and covariance Σ_x can then be computed by:

$$\begin{aligned} \mu_x &= A\mu_u + b \\ &= \nabla F(\mu_u)\mu_u + F(\mu_u) - \nabla F(\mu_u)\mu_u \\ &= F(\mu_u) \end{aligned}$$

$$\begin{aligned} \Sigma_x &= A\Sigma_u A^T \\ &= \nabla F(\mu_u)\Sigma_u \nabla F(\mu_u)^T \end{aligned}$$

Transformation of density functions III

Example: LRF Measurement Transformation

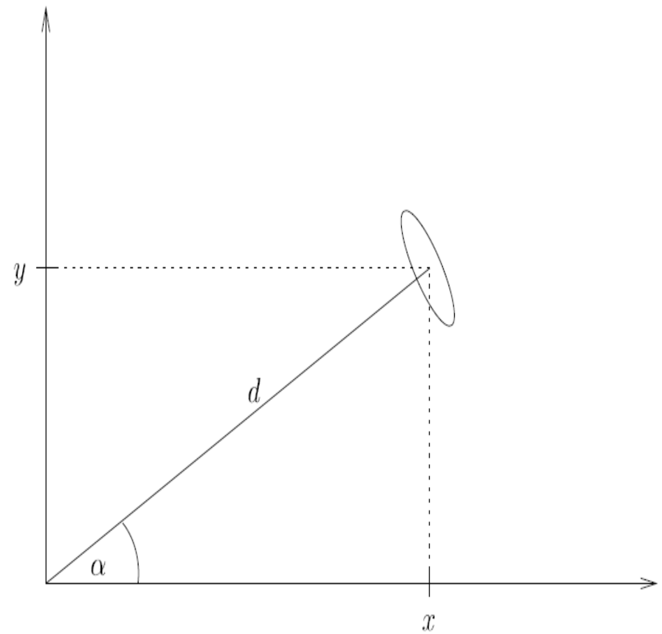
- We assume a normal distributed error of distance measurement *d* and angle measurement α:

$$d \sim N(\mu_d, \sigma_d^2) \qquad \alpha \sim N(\mu_\alpha, \sigma_\alpha^2)$$

- Transformation function F:

$$F(\begin{pmatrix} d \\ \alpha \end{pmatrix}) = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} d\cos\alpha \\ d\sin\alpha \end{pmatrix}$$

$$\begin{aligned} \Sigma_{xy} &= \nabla F_{d\alpha} \Sigma_{d\alpha} \nabla F_{d\alpha}{}^T \\ \nabla F_{d\alpha} &= \begin{pmatrix} \cos\alpha & -d\sin\alpha \\ \sin\alpha & d\cos\alpha \end{pmatrix} \\ \Sigma_{d\alpha} &= \begin{pmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\alpha^2 \end{pmatrix} \end{aligned}$$



Transformation of density functions IV

Example: LRF Measurement Transformation

- Assume d=3000mm, α=30°, σ_d=100mm, σ_α =5.7°

$$\begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix} = \begin{pmatrix} d\cos\alpha \\ d\sin\alpha \end{pmatrix} = \begin{pmatrix} 2598 \\ 1500 \end{pmatrix}$$

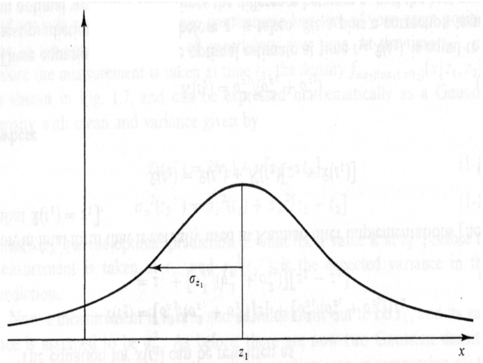
$$\nabla F_{d\alpha} = \begin{pmatrix} \cos\alpha & -d\sin\alpha \\ \sin\alpha & d\cos\alpha \end{pmatrix} = \begin{pmatrix} 0.866 & -1500 \\ 0.500 & 2598 \end{pmatrix}$$

$$\Sigma_{d\alpha} = \begin{pmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\alpha^2 \end{pmatrix} = \begin{pmatrix} 10000 & 0 \\ 0 & 0.01 \end{pmatrix}$$

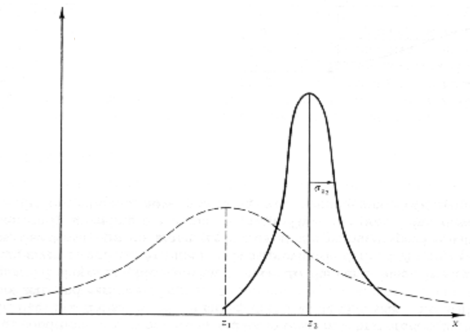
$$\Sigma_{xy} = \nabla F_{d\alpha} \Sigma_{d\alpha} \nabla F_{d\alpha}{}^T = \begin{pmatrix} 30000 & -34640 \\ -34640 & 70000 \end{pmatrix}$$

Kalman Filter I
Introduction

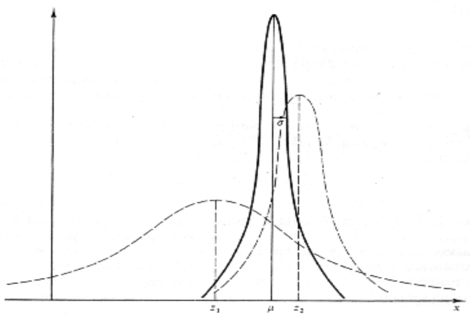
- An *optimal recursive data processing algorithm*
  - *optimal* since it processes all data regardless of precision
  - *recursive* since the filter computes the next estimate based on the last estimate and the latest measurement
- Fusion of two independent measurements of the same concept
- Each measurement has a confidence expressed by the variance of the Gaussian
- Example: two people on a boat estimate their 1D location. The 2nd person (z_2) is more skilled than the 1st one (z_1)



Conditional density of observation z_1



Conditional density of observation z_2



Conditional density after combining z_1 and z_2

Images courtesy of Maybeck

Kalman Filter II

Update Formula

one-dimensional:

$$
\begin{aligned}
l_1 &\sim N(\mu_1, \sigma_1^2) \\
l_2 &\sim N(\mu_2, \sigma_2^2)
\end{aligned}
$$

$$
\begin{aligned}
l &= \frac{1}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}} \left(\frac{1}{\sigma_1^2} l_1 + \frac{1}{\sigma_2^2} l_2\right) \\
\frac{1}{\sigma^2} &= \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}
\end{aligned}
$$

n-dimensional:

$$
\begin{aligned}
m_1 &\sim N(\mu_1, \Sigma_1) \\
m_2 &\sim N(\mu_2, \Sigma_2)
\end{aligned}
$$

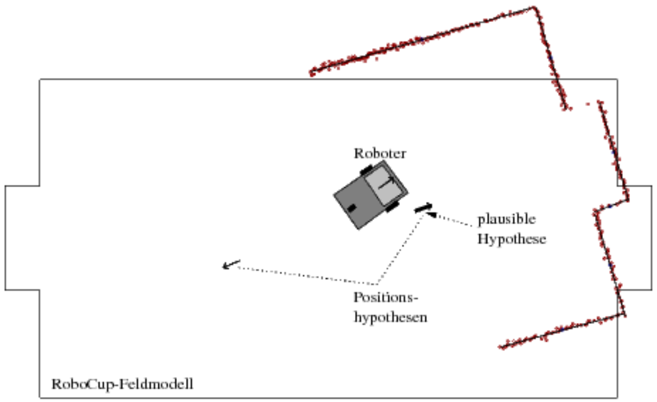
n-dimensional vector

nXn matrix

$$
\begin{aligned}
m &= (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1} (\Sigma_1^{-1} m_1 + \Sigma_2^{-1} m_2) \\
\Sigma &= (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}
\end{aligned}
$$

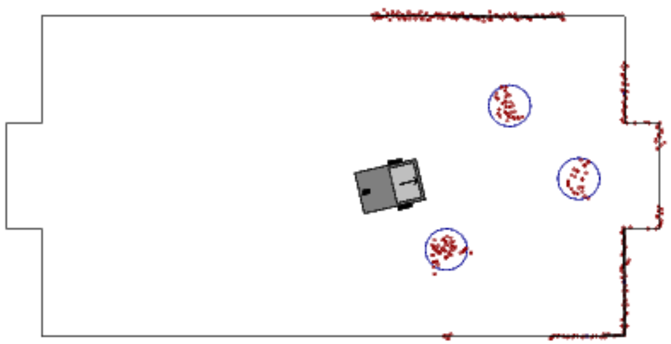
Case-Study: Cooperative opponent sensing on a soccer field I

1st step: players estimate their own position and orientation on the field by matching scans with the field model



Matching a scan to field model

2nd step: Extraction of other players by discarding scan points belonging to field walls and clustering the remaining ones. For each cluster the center of gravity is assumed to correspond to the center of another robot.

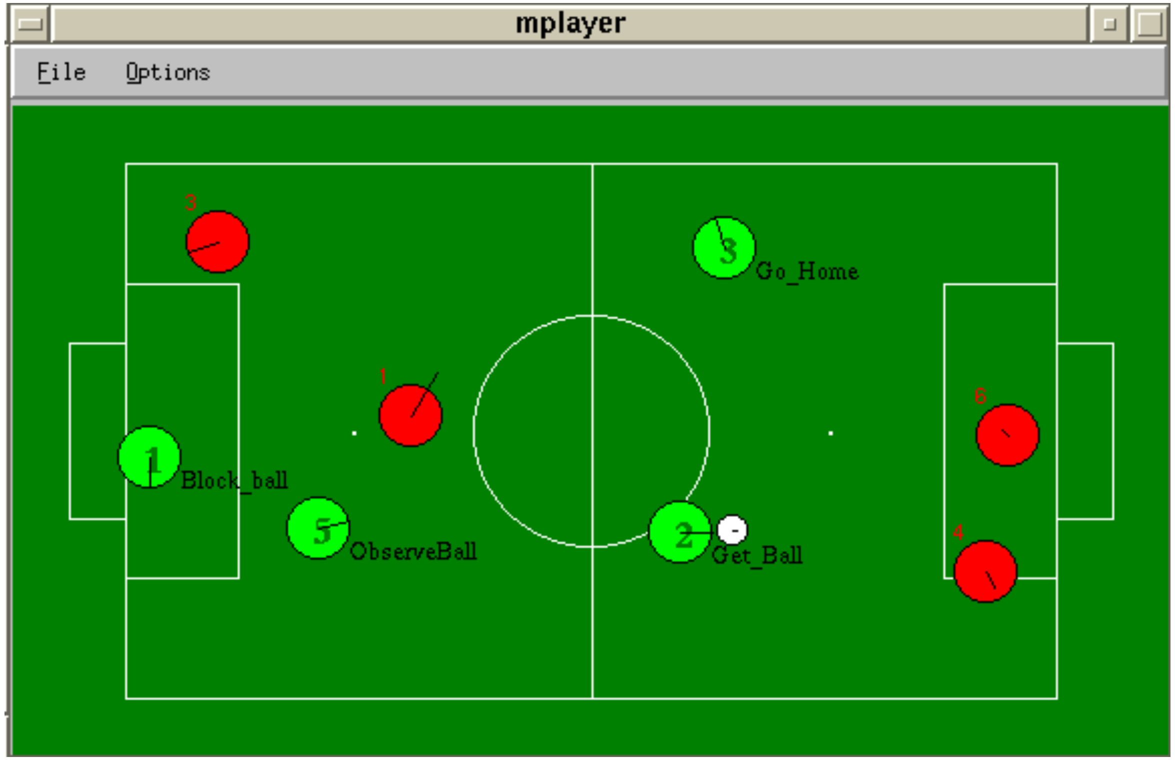


Extracting and clustering objects

Case-Study: Cooperative opponent sensing on a soccer field II

3rd step: Communication of position, heading and velocity of each detected object and own pose to a central multi-sensor integration module

4th step: Assignment of team player IDs to objects → Detection of opponents (red)



Cooperative world model

Integration of multiple measurements I

State representation:

Each observation is modeled by a random variable: $\mathbf{x}_s = (x_s, y_s, \theta_s, v_s, \omega_s)^T$

With mean $\hat{\mathbf{x}}_s$ and covariance Σ_s , where (x_s, y_s) is the position, θ_s the orientation, and v_s , ω_s are the translational and rotational velocities* of the object.

Modeling of the covariance: $\Sigma_s = diag(\sigma^2_{x_s}, \sigma^2_{y_s}, \sigma^2_{\theta_s}, \sigma^2_{v_s}, \sigma^2_{\omega_s})$,

where $\sigma_{x_s}, \sigma_{y_s}, \sigma_{\theta_s}, \sigma_{v_s}\ \sigma_{\theta_s}$ are constant standard deviations determined experimentally

State projection:

$$\hat{\mathbf{x}}_r \leftarrow F_s(\hat{\mathbf{x}}_r, t) = \begin{pmatrix} \hat{x}_r + \cos(\hat{\theta}_r)\hat{v}_r t \\ \hat{y}_r + \sin(\hat{\theta}_r)\hat{v}_r t \\ \hat{\theta}_r + \hat{\omega}_r t \\ \hat{v}_r \\ \hat{\omega}_r \end{pmatrix}$$

Velocities are computed from position history

$$\Sigma_r \leftarrow \nabla F_s \Sigma_r \nabla F_s^T + \Sigma_a(t)$$

$$\Sigma_a(t) = diag(\sigma^2_{x_a} t, \sigma^2_{y_a} t, \sigma^2_{\theta_a} t, \sigma^2_{v_a} t, \sigma^2_{\omega_a} t)$$

Constant standard deviations determined experimentally

***Note velocities are determined by differencing the last 10 pose estimates**

Integration of multiple measurements II

State update:

(a) Observation of a new object:

$$\hat{\mathbf{x}}_r = \hat{\mathbf{x}}_s, \qquad \Sigma_r = \Sigma_s$$

(b) Observation of a known object:

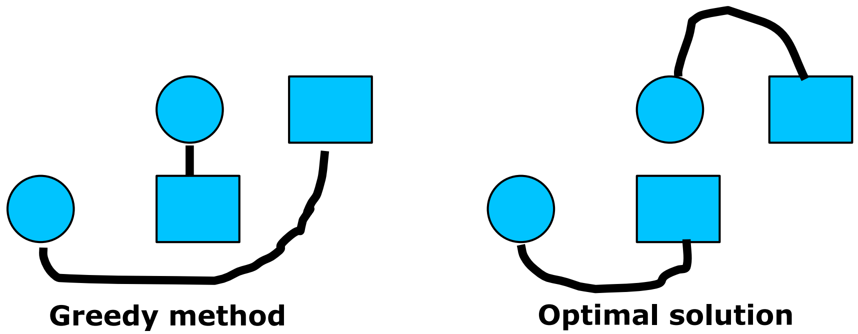
$$\hat{\mathbf{x}}_r \leftarrow (\Sigma_r^{-1} + \Sigma_s^{-1})^{-1}(\Sigma_r^{-1}\hat{\mathbf{x}}_r + \Sigma_s^{-1}\hat{\mathbf{x}}_s)$$
$$\Sigma_r \leftarrow (\Sigma_r^{-1} + \Sigma_s^{-1})^{-1}$$

Data association problem, i.e. how to associate observations to known objects?

Greedy method:

Search the global world model for the track whose predicted mean is closest to the observation. Assign observation if distance is beyond a certain threshold.

→ Can be sub-optimal!



Better approach: *geometric assignment*

Go over all possible sets of assignment pairs (s_i, r_i)

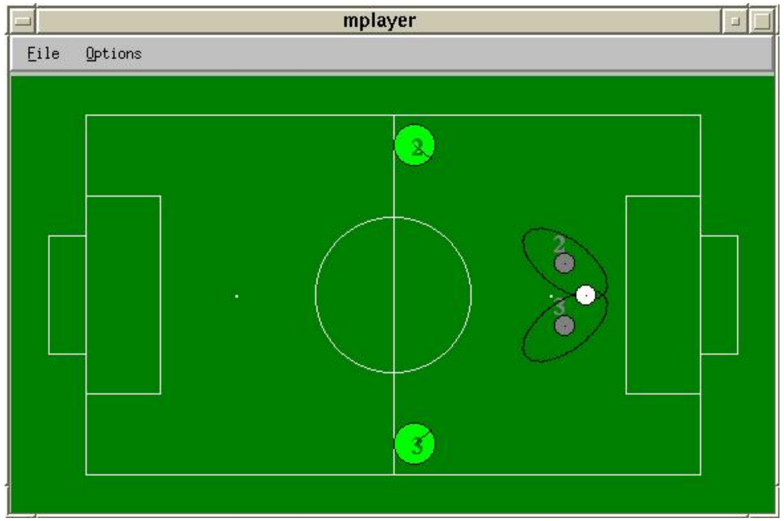
Find assignment that minimizes $\sum_{i=1}^{n} dist(s_i, r_i)^2$

Single Object Tracking from Noisy Data
Example: Ball Tracking

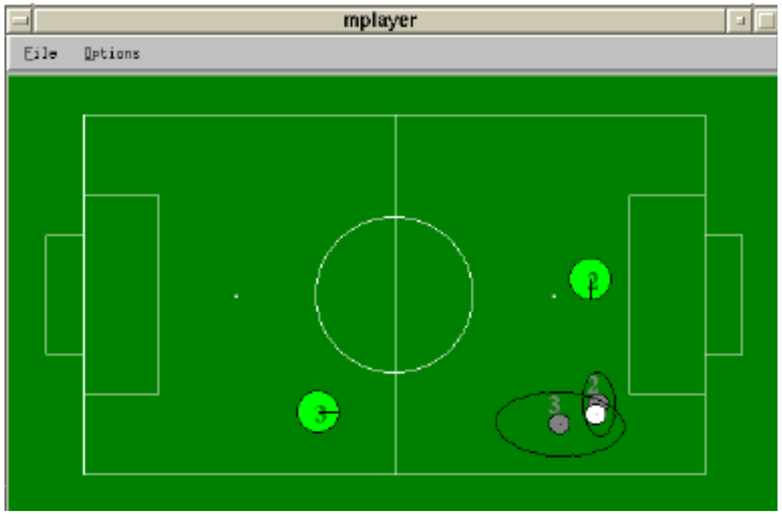
- For example, global ball position estimation: stereo vision with robot groups
- Detection of the ball by vision, e.g. detecting the ball by color
  - Estimation of the angle is quite accurate, however, distance is not
  - Kalman Filter integration yields an error ellipse with respect to these confidences
  - Fusion of two estimates respects error ellipse: effect of “triangulation”
- Prediction step (predict next location where ball will be observed):
  - Project ball position into the future using a constant negative ball acceleration (due to friction)
  - Consider a certain projection error
- Update step (when new observation is made):
  - Integrate new measurement (using a weighted average on the error)
    - distance error grows with distance
    - angular error is small and constant

Single Object Tracking from Noisy Data

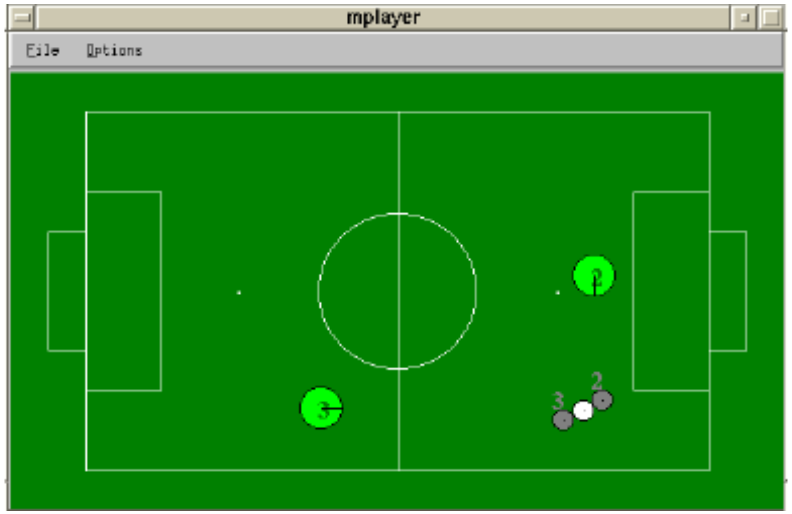
Example: Ball Tracking



Effect of triangulation



Kalman filtering



Simple averaging

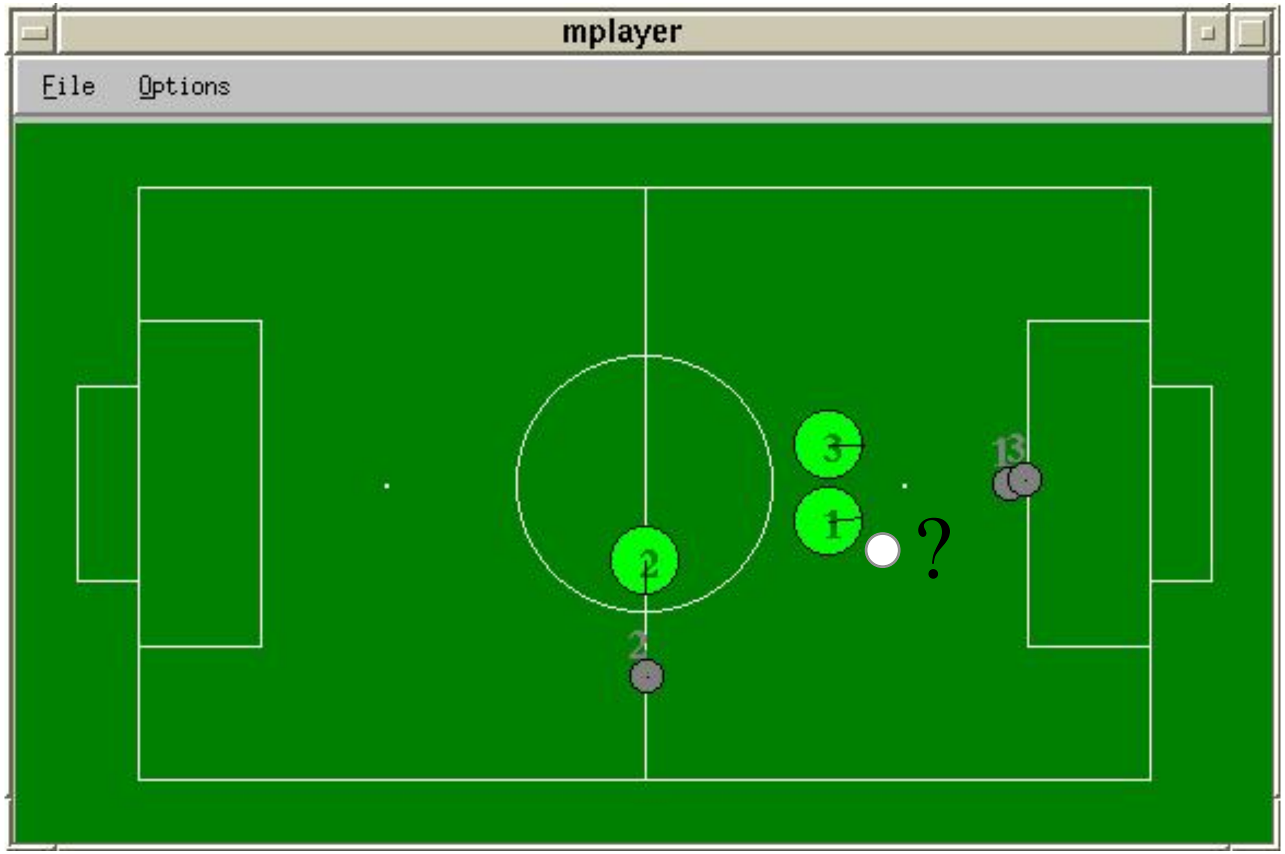
Kalman filtering compared to simple averaging: highly confident estimates are more strongly weighted

The Importance of Global Ball Estimation

Minho (Portugal) shoots at our goal from the other side of the field. Our goalie gets this information early on from his team mates and can easily defend

Single Object Tracking from Noisy Data

Problem of false positives (ghost balls)

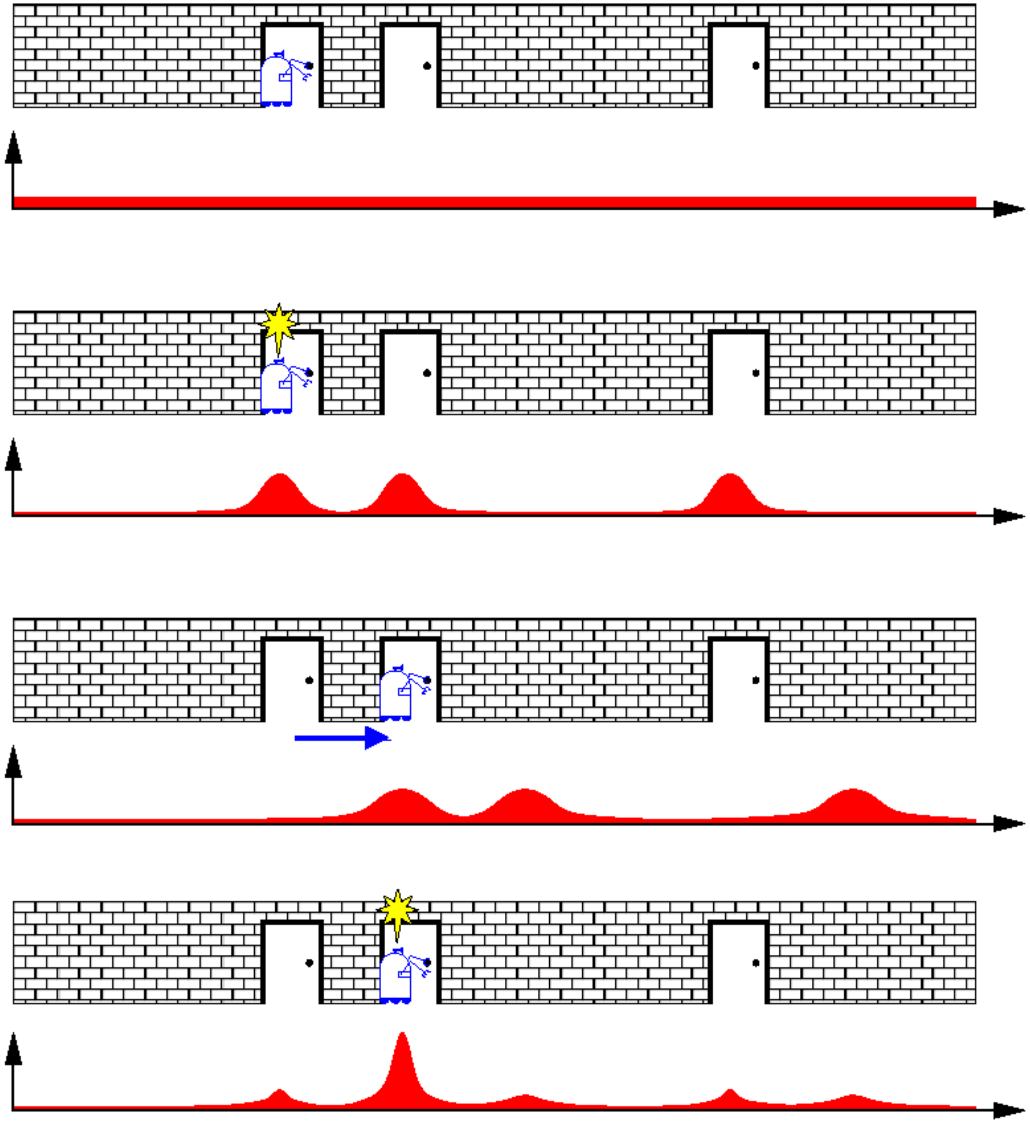


Player 2 is hallucinating

Markov Localization as Observation Filter
Introduction

- The Kalman-Filter expects that measurements originate from the same objects
  - However, color thresholding on a soccer field might confuse for example “red t-shirts” with the ball
  - Consequently, Kalman filtering yields poor results
- Markov localization: Simultaneous tracking of multiple hypotheses
- Idea: To filter-out false positives with a probability grid

Probabilistic Localization



Courtesy of Wolfram Burgard

Simple Example of State Estimation

- Suppose a robot obtains measurement z
- What is $P(open|z)$?

Courtesy of Wolfram Burgard

Causal vs. Diagnostic Reasoning

- *P(open|z)* is diagnostic.
- *P(z|open)* is causal.
- Often causal knowledge is easier to obtain.
- Bayes rule allows us to use causal knowledge:

$$P(open \mid z) = \frac{P(z \mid open)P(open)}{P(z)}$$

Courtesy of Wolfram Burgard

Example

- $P(z|open) = 0.6$ $\quad$ $P(z|\neg open) = 0.3$
- $P(open) = P(\neg open) = 0.5$

$$P(open \mid z) = \frac{P(z \mid open)P(open)}{P(z \mid open)p(open) + P(z \mid \neg open)p(\neg open)}$$

$$P(open \mid z) = \frac{0.6 \cdot 0.5}{0.6 \cdot 0.5 + 0.3 \cdot 0.5} = \frac{2}{3} = 0.67$$

z raises the probability that the door is open.

Courtesy of Wolfram Burgard

Markov Localization as Observation Filter
Prediction & Update I

- Discretization of the soccer field into a two-dimensional grid
  - Each cells of the grid reflects the probability p(z) that the ball is at the cell location loc(z) = (x,y)
- Uniform initialization of the grid before any observation is processed, i.e. p(z)=1/#cells
- Prediction step:
  - Simple model of ball motion $p(z) \leftarrow \sum p(z \mid z')p(z')$ here $p(z \mid z')$ denotes the probability that the ball moved to cell z given it was at z'.
  - When assuming that all kind of motion directions are equally possible, and velocities are normally distributed with zero mean and covariance σ_v^2, $p(z \mid z')$ can be modeled by a time-depended Gaussian around z':

$$p(z \mid z') \sim N(z', diag(\sigma_v^2 t, \sigma_v^2 t))$$

Markov Localization as Observation Filter
Prediction & Update II

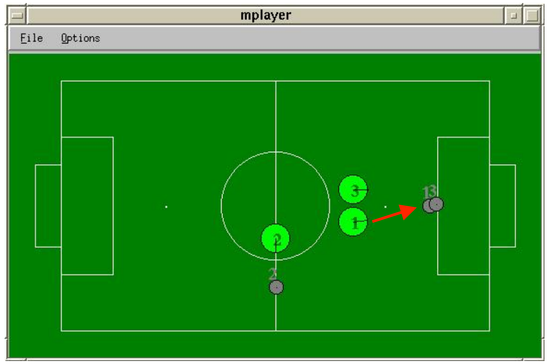
- Update step:
  - Fusion of new ball observation z_b into the grid according to Bayes' law:

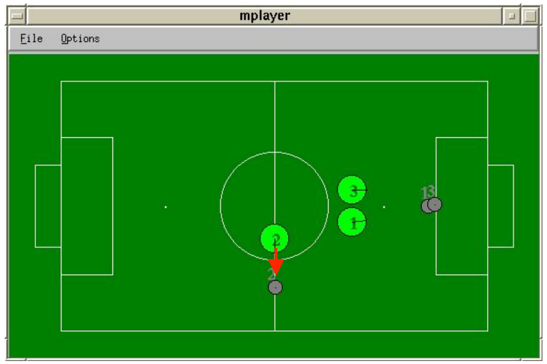
$$p(z) \leftarrow \frac{p(z_b \mid z)p(z)}{\sum_{z'} p(z_b \mid z')p(z')}$$

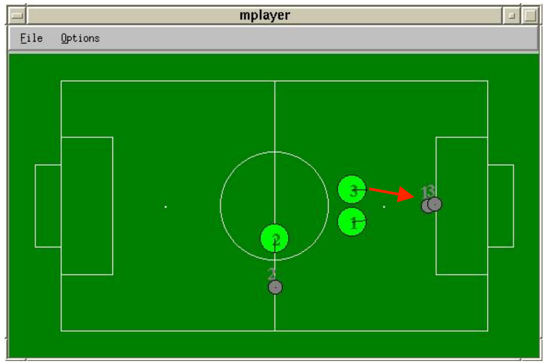
Normalization: Ensuring that probabilities sum up to 1.0

  - The sensor model $p(z_b \mid z)$ determines the likelihood observing z_b given the ball is at position z.
    - e.g. less confidence as more far away the ball
- Finally, the Markov grid can be used for outlier rejection
  - Kalman filtering is only applied at the highest peak of the distribution
  - If another peak becomes more likely, the Kalman filter is re-initialized accordingly

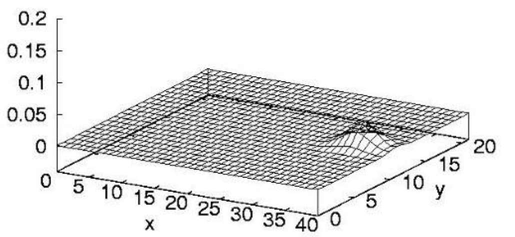
Phantom Balls: Development of Probability Distribution I



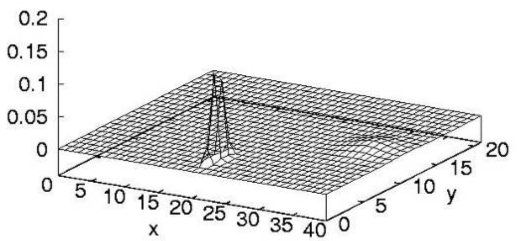




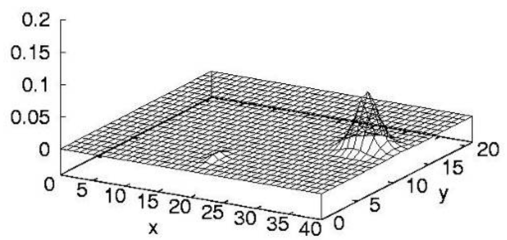
after 1st measurement (1)



after 2nd measurement (2)

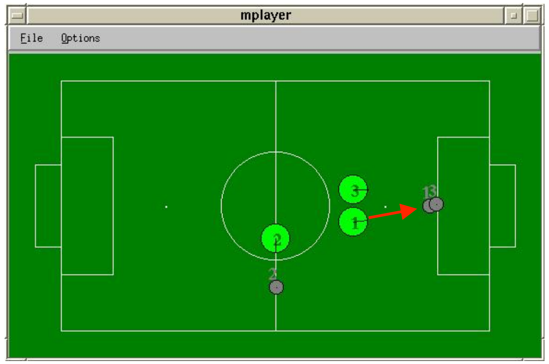


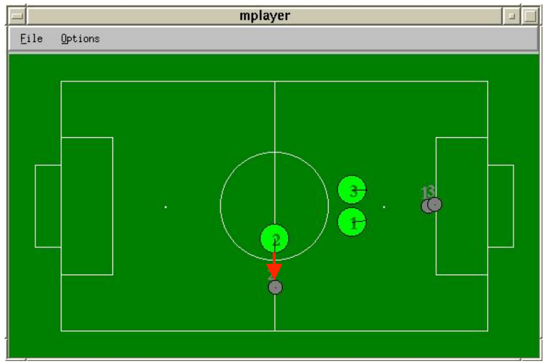
after 3rd measurement (3)

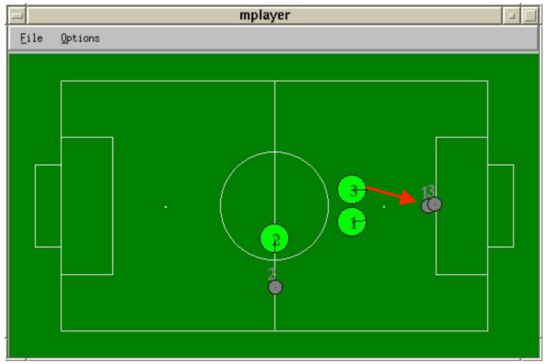


Consider area with highest peak as possible ball area
and use KF there

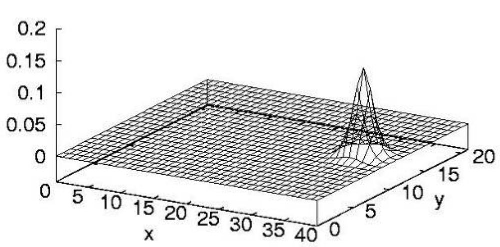
Phantom Balls: Development of Probability Distribution II



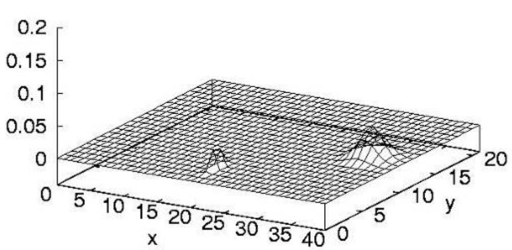




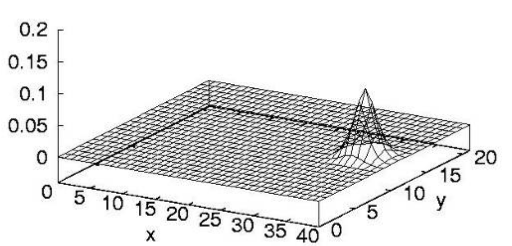
after 4th measurement (1)



after 5th measurement (2)



after 6th measurement (3)



At *RoboCup 2000*, 938 out of 118388 **(0.8%)** ball observations were ignored because of the Markov localization filter.

Demo Webplayer

See www.cs-freiburg.de

Potential Fields
Introduction

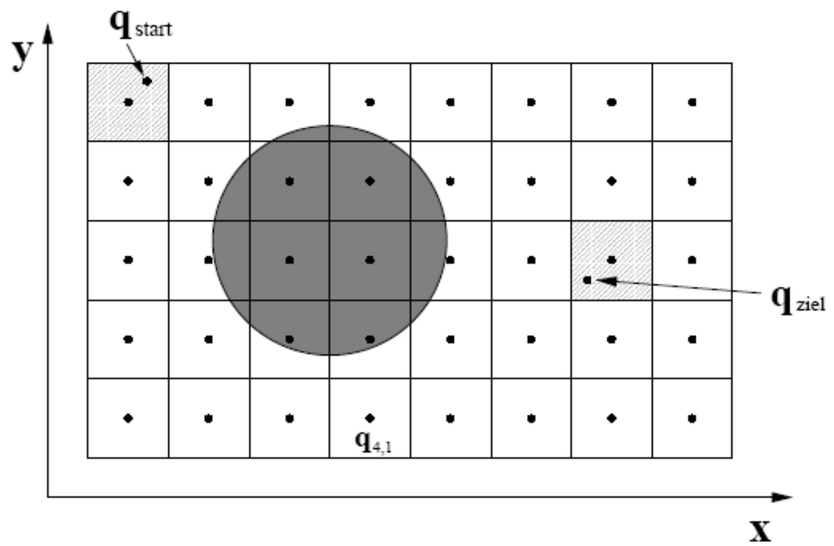
- Originally introduced for robot path planning
  - Robot is considered as particle within a force field, the potential field
  - Potential field is generated by overlaying repulsive potentials (e.g. obstacles) and attractive potentials (e.g. goals)
  - The motion of the robot is determined by negating the field's gradient, leading to the potential minimum
  - Repulsive and attractive potentials are computed separately
- Can also be used for strategic decision making (e.g. CS-Freiburg)

Potential Fields
Grid representation

- Discretization of the configuration space into equally sized cells
- Grid representation GC is defined for every **q**=(x,y) as follows:

$$\mathcal{GC} = \{\mathbf{q} \in \mathcal{C} \mid \mathbf{q} = (i\delta_x, j\delta_y),\ i = 1,\ldots,N,\ j = 1,\ldots,M\}$$

where δ_x, δ_y are the step sizes in X and Y direction, and N, M are the number of cells along the axes, respectively



Potential Fields

Potentials

- Potentials are differentiable functions of the type $\mathbf{U} : \mathcal{C}_{free} \rightarrow \mathbf{R}$, where C_{free} is the set of possible robot configurations
- Typically, high values indicate obstacles and low values goals
- Given differentiable potentials, one can compute the force at each configuration q by:

$$\vec{F}(\mathbf{q}) = -\vec{\nabla}\mathbf{U}(\mathbf{q})$$

- For example, given a 2D work space, force F(q) can be computed from U(q) by:

$$\vec{F}(\mathbf{q}) = -\begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{pmatrix} \mathbf{U}(\mathbf{q}) = -\begin{pmatrix} \frac{\partial \mathbf{U}(\mathbf{q})}{\partial x} \\ \frac{\partial \mathbf{U}(\mathbf{q})}{\partial y} \end{pmatrix}$$

Potential Fields
Attractive Potential

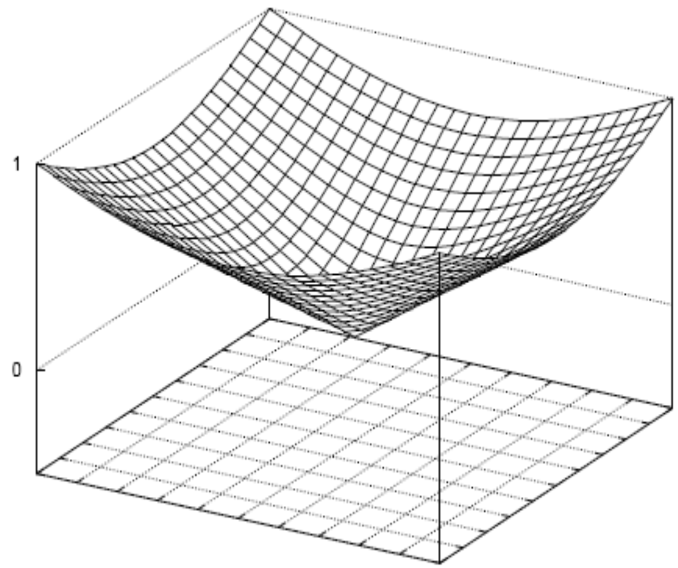
- Influence of potential has to be workspace wide!
- Linearly decreasing potential with increasing distance to goal q_{ziel}:

Euclidian distance

$$\mathbf{U}_{att}(\mathbf{q}) = \xi\rho(\mathbf{q}), \text{ with } \rho(\mathbf{q}) = \|\mathbf{q} - \mathbf{q}_{ziel}\| \text{ and scaling factor } \xi.$$

Computation of force F_{att}:

$$\begin{aligned} \vec{F}_{att}(\mathbf{q}) &= -\xi\vec{\nabla}\rho(\mathbf{q}) \\ &= -\xi(\mathbf{q} - \mathbf{q}_{ziel})/\|\mathbf{q} - \mathbf{q}_{ziel}\| \end{aligned}$$



Attractive potential

- Singularity at $q=q_{ziel}$!
  - Has to be dealt with separately

Potential Fields
Repulsive Potential

- Influence of potential can be limited in order to simplify computations
- Increasing potential with increasing distance to object:

Strong rep. pot. Close to the object!

$$\mathbf{U}_{rep}(\mathbf{q}) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0}\right)^2 & falls\ \rho(\mathbf{q}) \leq \rho_0 \\ 0 & falls\ \rho(\mathbf{q}) > \rho_0 \end{cases}$$

Where η is a scaling factor, p(**q**) the distance to the obstacle, and p_0 the maximal influence radius of the potential

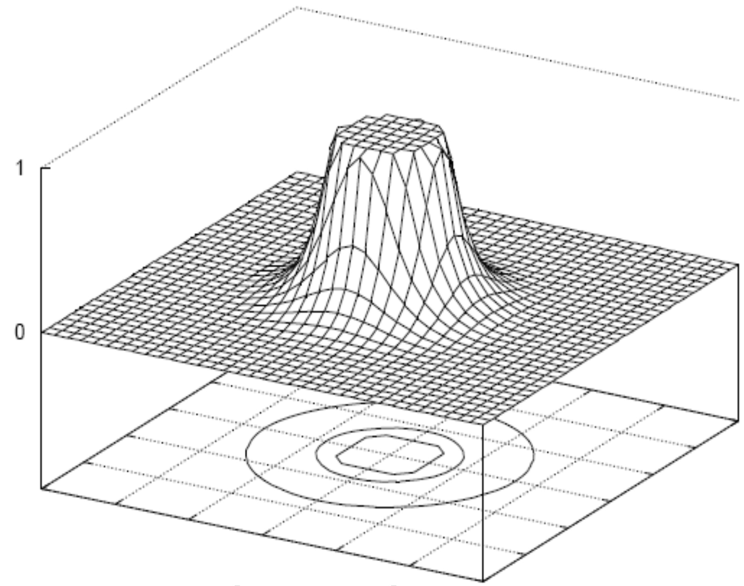
The distance function should respect the shape of the object, for example:

$$\rho(\mathbf{q}) = \min_{\mathbf{q}' \in \mathcal{CB}} \|\mathbf{q} - \mathbf{q}'\|$$

q' are all cells covered by the object

Computation of force F_{rep}:

$$\vec{F}_{rep}(\mathbf{q}) = \begin{cases} \eta\left(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(\mathbf{q})}\vec{\nabla}\rho(\mathbf{q}) & falls\ \rho(\mathbf{q}) \leq \rho_0 \\ 0 & falls\ \rho(\mathbf{q}) > \rho_0 \end{cases}$$

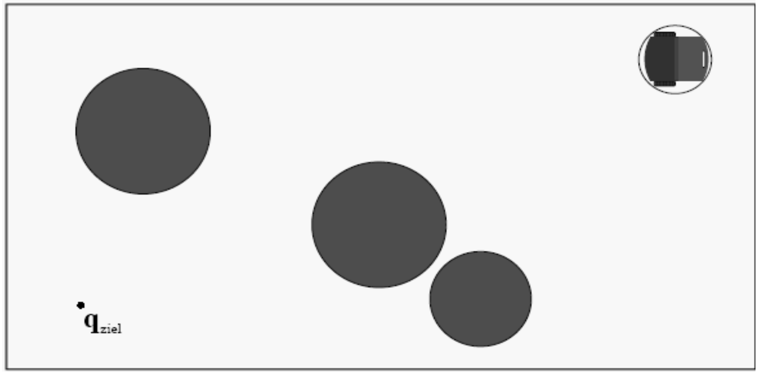


Repulsive potential

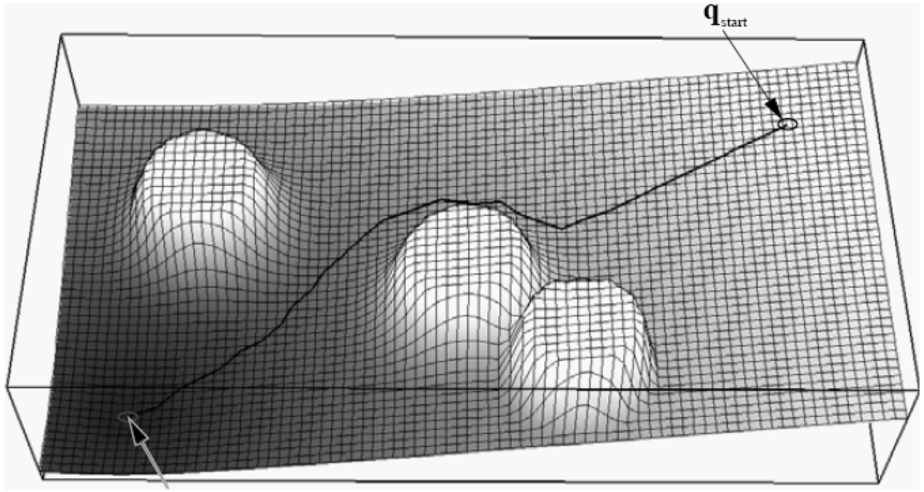
Potential Fields

Computing the potential field

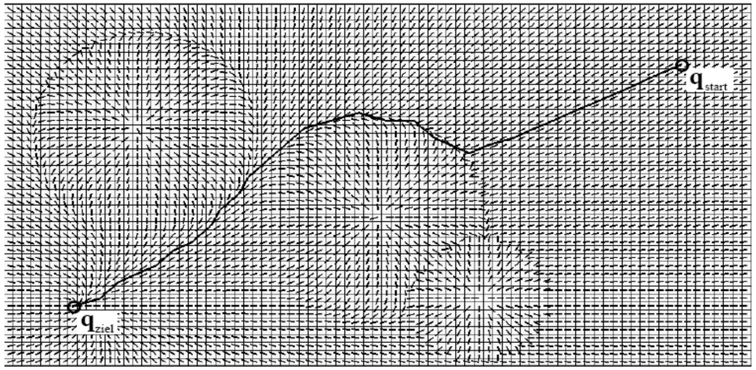
Computation by: $$\mathbf{U}(\mathbf{q}) = \sum_{i=1}^{M} \mathbf{U}_{att_i}(\mathbf{q}_{goal_i}) + \sum_{j=1}^{N} \mathbf{U}_{rep_j}(\mathbf{q})$$



Situation: Obstacles, start, and goal



Resulting potential field



Resulting force field

However: Reactive path selection can lead into local minima! Better: Finding goal with A* and using Potential Field values as heuristic.

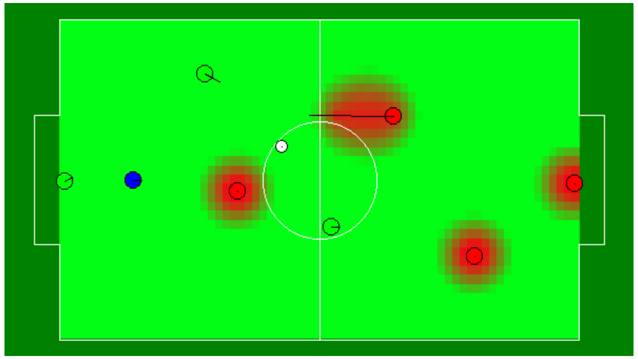
Case-Study: Extracting predicates for playing soccer I

- Predicates are the basis for action selection and strategic decision making
- Can be considered as world model abstractions
- Simple predicates of objects (can be directly computed from positions):
  - ***InOpponentsGoal****(object)*
    - Object in opponent goal?
  - ***InOwnGoal****(object)*
    - Object in own goal?
  - ***CloseToBorder****(object)*
    - The distance to any border is beyond a threshold?
  - ***FrontClear****()*
    - Neither another object nor the border is in front?
  - ***InDefense****(object)*
    - Object in the last third of the soccer field?

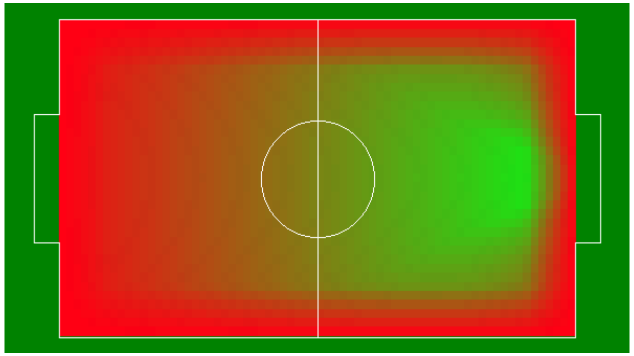
Case-Study: Extracting predicates for playing soccer II

- Extended predicates:
  - computed by normalized potential fields: (f_i: $\Re \times \Re \rightarrow [0..1]$)
  - discretized by grid, e,g., 10x10cm cell size

- Examples:
  - f_{free}: indicates positions under the influence of the opponent
  - $f_{covered}$: indicates position covered by teammates
  - $f_{desired}$: indicates tactical good positions



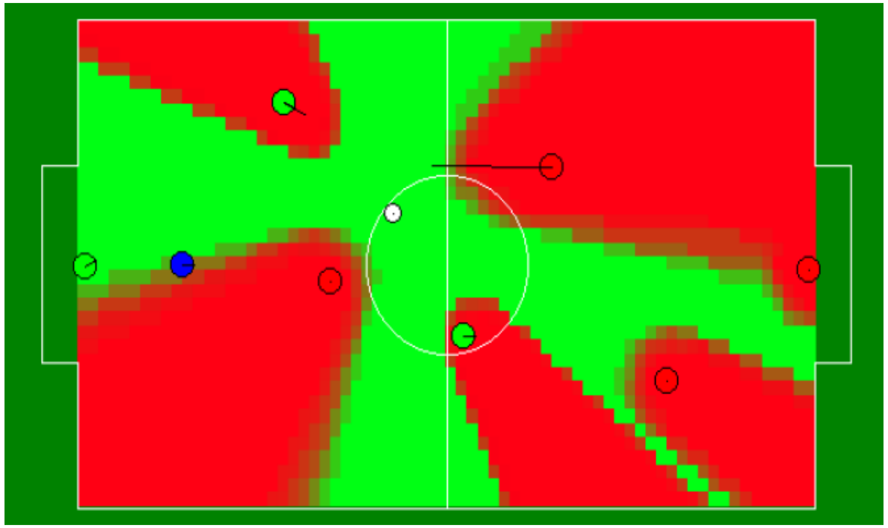
f_{free}



$f_{desired}$

Case-Study: Extracting predicates for playing soccer III

- Given the predicates, for each role and action the best next desired position can be computed
- Combined potential fields:
  - $f_{ballview}$: indicates whether the direct line from the ball to a position is free
  - Recursive computation:



$$f_{ballview}(k(z_1)) = 1$$
$$f_{ballview}(k(z_i)) = f_{ballview}(k(z_{i-1})) \cdot f_{free}(k(z_i)) \cdot (1 - f_{covered}(k(z_i)))$$

Where $z_1,\ldots,z_n$ are the indices of “lines”, i.e., the cells going from the ball towards the border (star-like)

Summary

- Consistent world models are the key to deliberative acting!
- The Kalman Filter is a tool for accurately estimating object poses
  - However, only single hypotheses can be tracked
- Markov Localization is a tool for robust object tracking by considering multiple hypotheses
  - However, accuracy depends on the choosen discretization
- Best results are yielded by combining both methods
- Potential Fields are an efficient tool for generating predicates from complex representations, simplifying decision making of a mobile agent
- For path planning, care has to be taken on local minima

Literature

- Kalman Filter:
  - Peter Maybeck ***Stochastic Models, Estimation, and Control***, Volume 1, *Academic Press*, Inc.
  - Website: http://www.cs.unc.edu/~welch/kalman/
- Markov Localization:
  - S. Thrun, W. Burgard, and D. Fox **Probabilistic Robotics**. *MIT-Press*, 2005.
- Cooperative Sensing:
  - M. Dietl, J.-S. Gutmann and B. Nebel, **Cooperative Sensing in Dynamic Environments**, in: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'01)*, Maui, Hawaii, 2001
  - M. Dietl, ***Globale Sensorfusion im Roboterfußball***, Studienarbeit, Fakultät für Angewandte Wissenschaften, Univ. Freiburg, Dez. 1999
- Potential Fields:
  - T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner and B. Nebel **CS-Freiburg: Coordinating Robots for Successful Soccer Playing** *IEEE Transactions on Robotics and Automation* 18(5):685-699, October 2002
  - C. Reetz, ***Aktionsauswahl in dynamischen Umgebungen am Beispiel Roboterfußball***, Diplomarbeit an der Fakultät für Angewandte Wissenschaften, Univ. Freiburg, 1999
  - T. Weigel, J.-S. Gutmann, B. Nebel, K. Müller, and M. Dietl, **CS Freiburg: Sophisticated Skills and Effective Cooperation**, *Proc. European Control Conference (ECC-01)*, Porto, Portugal, September 2001