Localization in Mobile Robotics

Part I.

Miroslav Kulich

Intelligent and Mobile Robotics Group
Czech Institute of Informatics, Robotics and Cybernetics
Czech Technical University in Prague

Winter semester 2020/2021

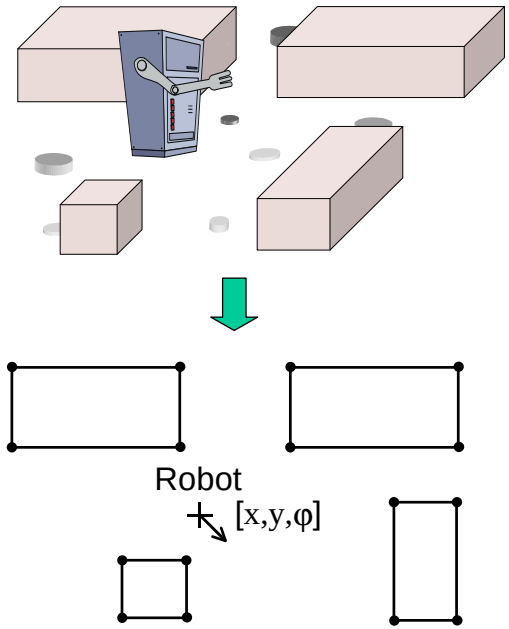
last update: 23rd Sept, 2020

Lecture outline

- Taxonomy
- Continuous localization
  - Iterative Closest Point
  - Iterative Matching Range Point
  - Iterative Dual Correspondence
  - Correlative Scan Matching
- Probabilistic methods
  - Bayes filter
  - Particle filter
  - Histogram filter
  - Kalman filter
  - Extended filter

Localization

- Localization is the problem of robot position estimation
  - relatively with respect to the start robot position
  - in the given coordinate system (in a known map)
- Pose is supposed also with orientation: i.e. (x, y, ϕ), resp. $(x, y, z, roll, pitch, yaw)$
- Input: odometer information, data measurements



Why is localization difficult?

- Position can be measured with no single sensor $=>$ position is determined from sensory data
- But sensors are inaccurate (subject of noise)!
- Control commands are realized inaccurately.
- One single measurement is not enough.

Taxonomy

- Position tracking (continuous, local)
  - The initial position is known.
  - Correction of odometer inaccuracies.
  - The error is bounded.
  - Unimodal distribution (Gaussian).
- Lost robot problem(global, absolute)
  - The initial position is not known.
  - The error is unbounded.
  - Multi-modal distribution.
- kidnapped robot problem
  - Detection of position errors $=>$ correction.
  - Suitable for testing.

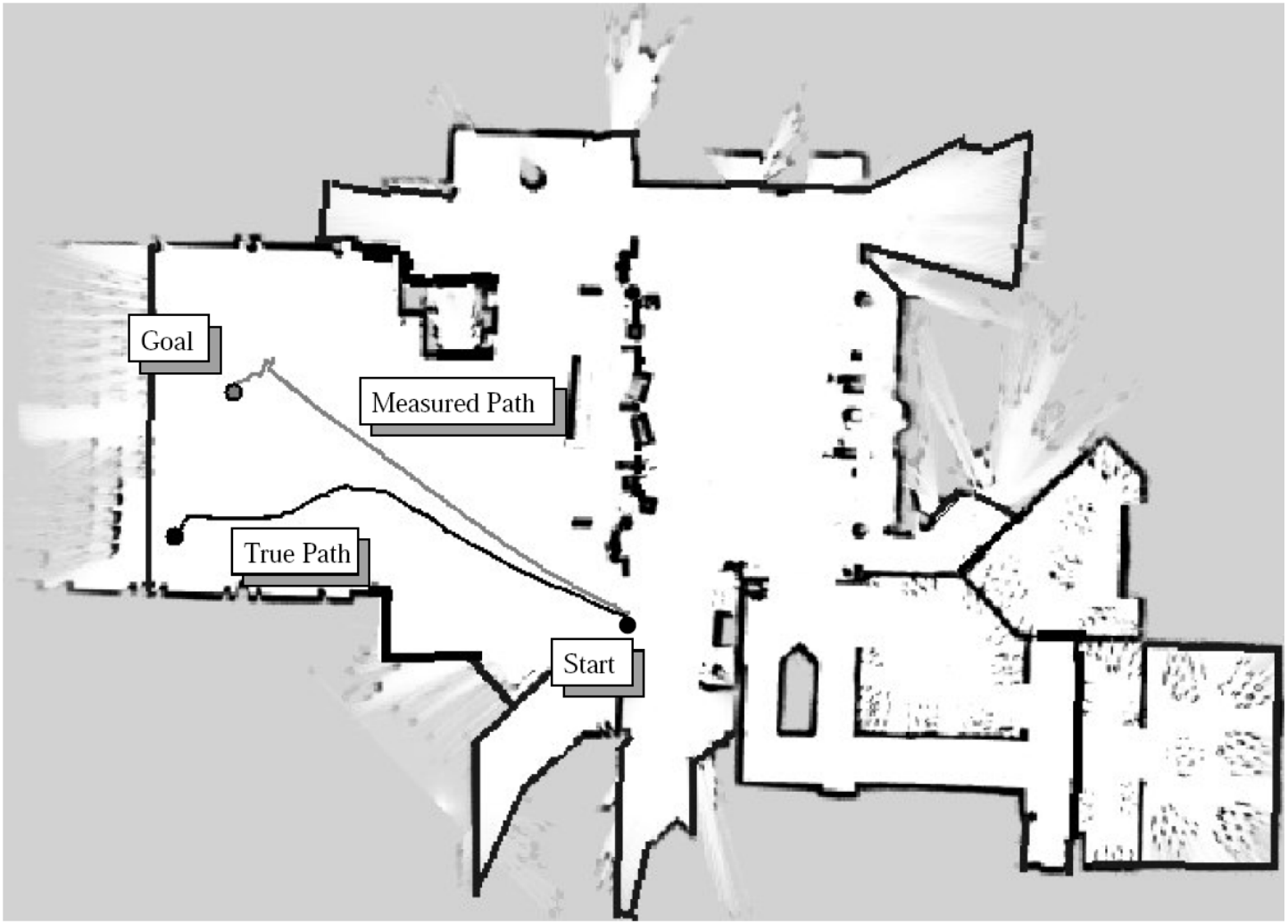
Taxonomy

- Static environment
  - The robot is the only one what moves.
  - The robot position is the only variable.
  - "Nice" mathematical features.
- Dynamic environment
  - Objects or/and other robots move (change their positions/state).
  - Objects: humans, light (for cameras), doors, ...
  - Two approaches
    - Object's movement can be modeled => it is a part of state description of the localization task.
    - Data about moving objects are filtered.

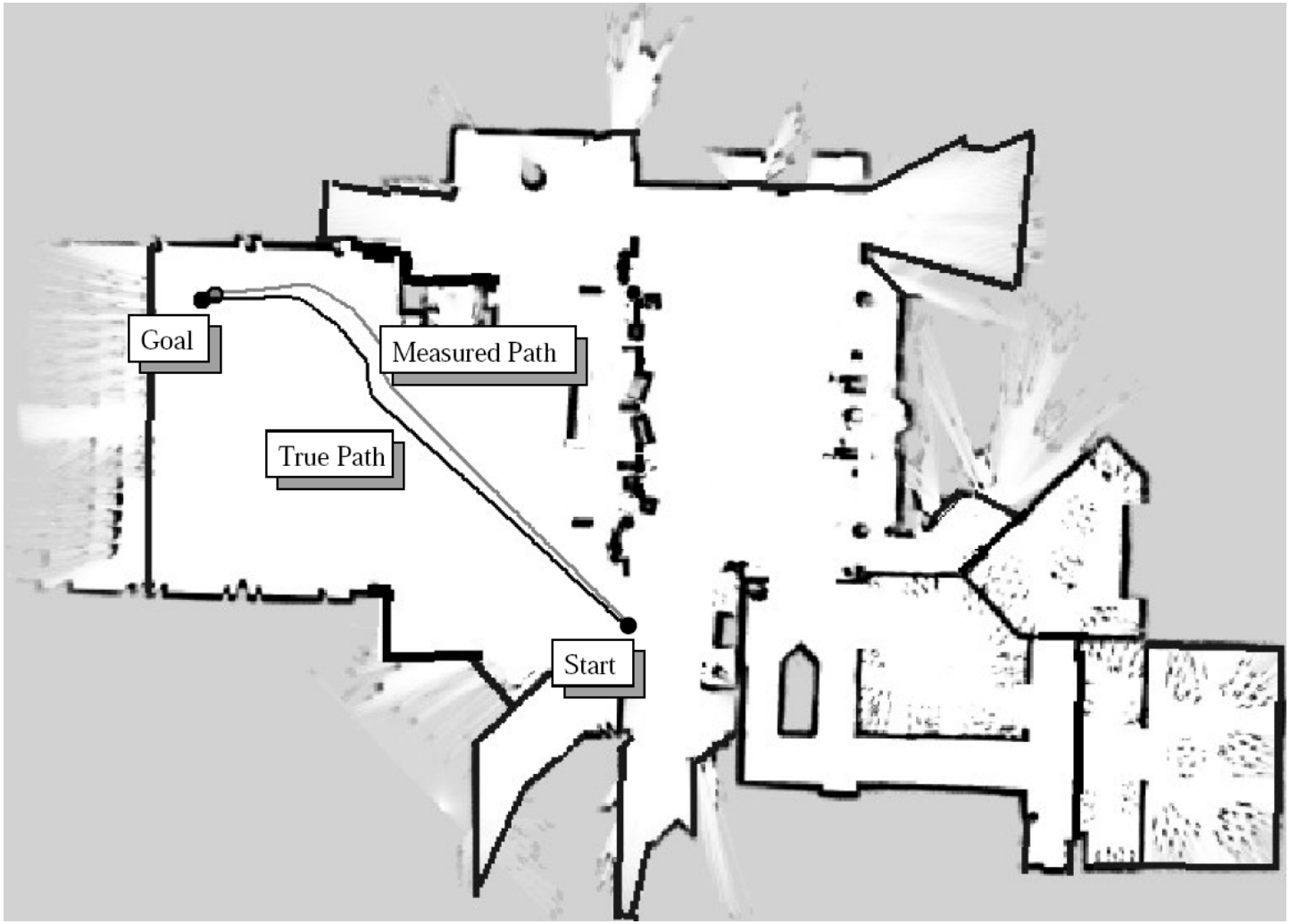
Taxonomy

- Passive localization
  - Localization module only passively observes, what is happening - it has no influence on the robot control.
  - Localization is a “complementary product of other task.
- Active localization
  - The robot is controlled in order to correct/determine its position.
  - Gives better results . . . .
  - . . . but it can’t be applied every time.
  - Goals alternation: the robot is localized sometimes x the primary task is solved sometimes.
  - The goal is generated as a compromise.

Passive localization



Active localization

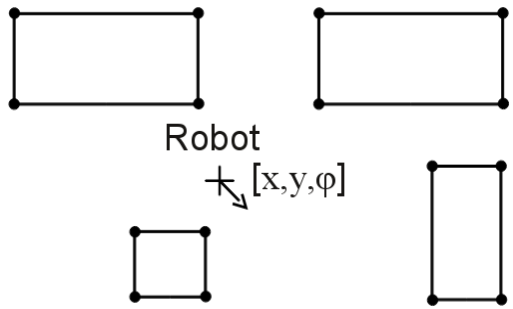


Taxonomy

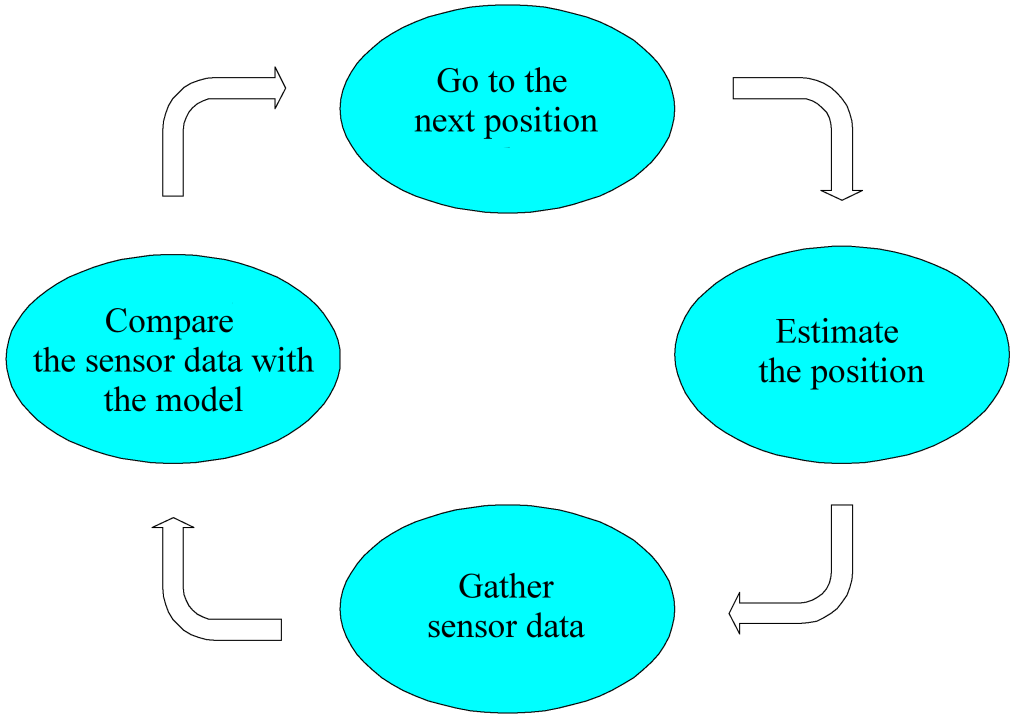
- Single robot
  - The classical problem.
  - No communication is need, everything at one place.
- Multiple robots
  - Every robot is localized separately/independently (it pretends that other robots do not exist).
  - But if the robots are able communicate or detect each other, it is better.
  - Self-localization in the other robot map.
  - My position estimate is known, other's robot position estimates are known as well as relative positions $=>$ optimization (rubber bands).

We will suppose

- A terrestrial robot operating on a horizontal plane (2D).
- Only one plane of the environment is sensed.
- Majority of objects is static and detectable by sensors
- Distance measuring sensors are used.
- Passive localization.
- Single robot.
- Continuous, absolute localization, and kidnapped robot

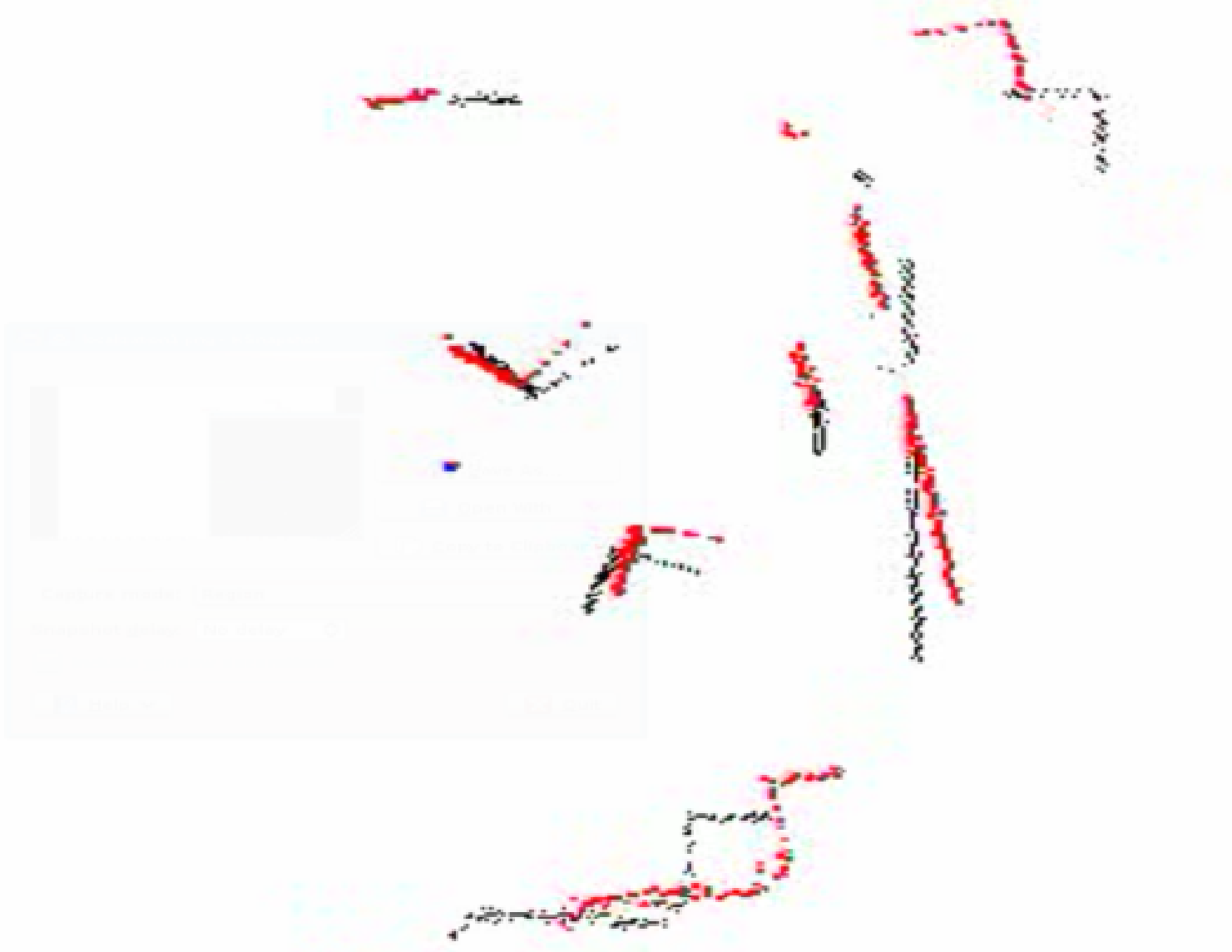


Continuous localization



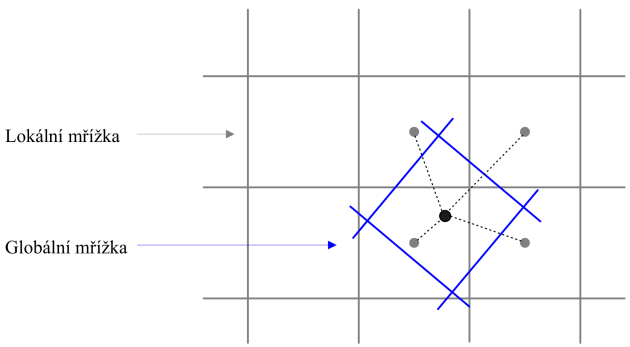
- The position is supposed to be in the close neighborhood of the estimate.
- High precision needed (continuous process).
- Input: local a and global maps (scans).
- Output: transformation minimizing measure of maps discrepancy.

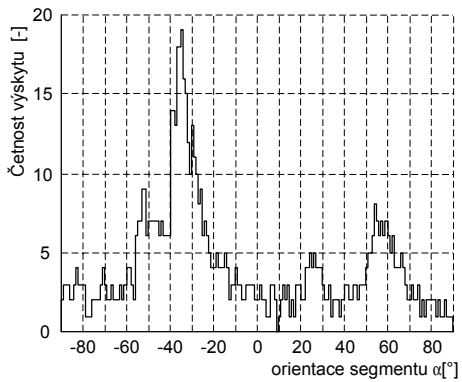
Example

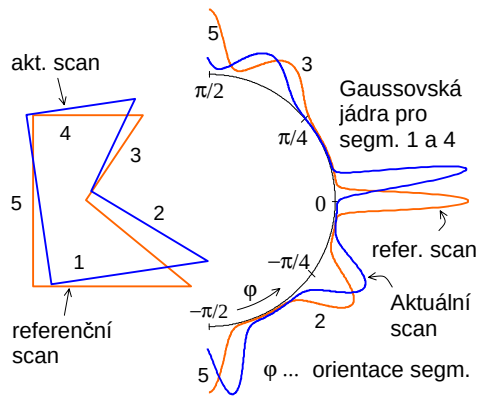


Continuous localization

- Many methods differing in the representation of the maps (scans)
- It is always optimization problem.
- Approaches:
  - Localization on occupancy grids.
  - Histograms
  - Localization based on processing of geometric primitives (point-to-line, line-to-line, Hough transform)





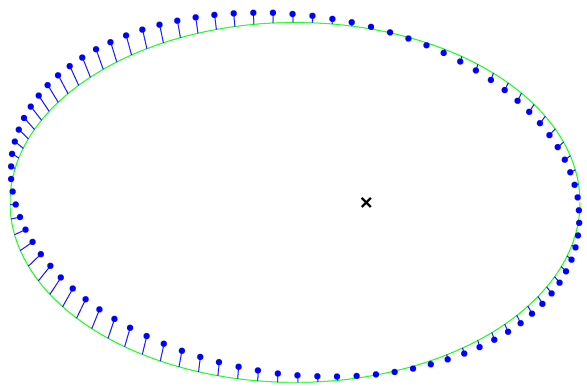


Iterative Closest Point (ICP)

(Lu, Milios)

- Iterative
  - The actual scan is interpolated with a polyline.
  - Corresponding points are determined (the smallest distance).
  - Transformation is determined (using least squares).

$$E_{dist}(T_x, T_y, \omega) = \sum_i^n |R_\omega P + T - P'|^2$$



- It has the analytical solution:

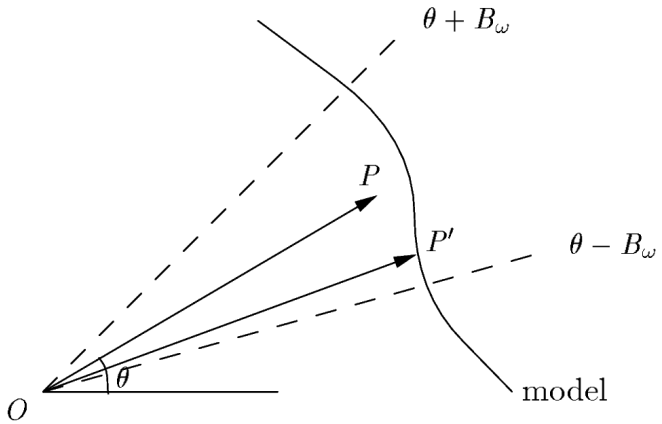
$$\begin{aligned} T_x &= \bar{x}' - (\bar{x}\cos\bar{\omega} - \bar{y}\sin\bar{\omega}) \\ T_y &= \bar{y}' - (\bar{x}\sin\bar{\omega} + \bar{y}\cos\bar{\omega}) \\ \bar{\omega} &= \arctan \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i' - \bar{y}') - \sum_{i=1}^n (y_i - \bar{y})(x_i' - \bar{x}')}{\sum_{i=1}^n (x_i - \bar{x})(x_i' - \bar{x}') + \sum_{i=1}^n (y_i - \bar{y})(y_i' - \bar{y}')} \end{aligned}$$

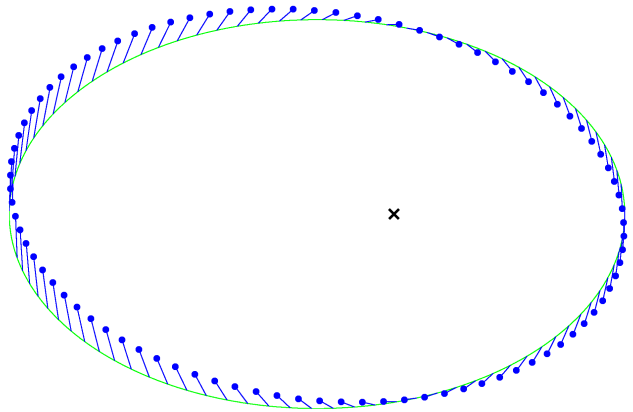
- It converges slowly, especially in the rotational part.

Iterative Matching Range Point (IMRP)

(Lu, Milios)

- The procedure is the same as in ICP, except the correspondence rule.
- Consider that translation is irrelevant $=>$ $|P| \approx |P'|$ and $\tilde{\phi} \approx \phi + \omega$.
- Neighborhood size for correspondence search is decreasing.
- Initially slower convergence than ICP, but it is faster later on.
- IMRP tries to correct translation error with rotation $=>$ instability.

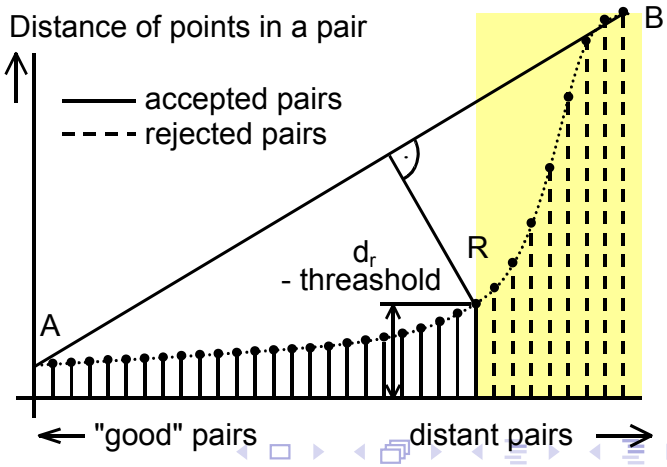




Iterative Dual Correspondence (IDC)

- Combination of the two rules: translation from ICP, rotation from IMRP
- Algorithm
  - For each P_i from the actual scan:
    - Apply the *closest-point* rule to determine the corr. point P'
    - Apply the *matching-range-point* rule $=>$ P''
    - Compute the least-squares solution (ω_1, T_1) from the set of correspondence pairs (P, P')
    - Compute the least-squares solution (ω_2, T_2) from the set of correspondence pairs (P, P'')
    - Form (ω_2, T_1) as the solution
  - Repeat the previous steps until the process converges.

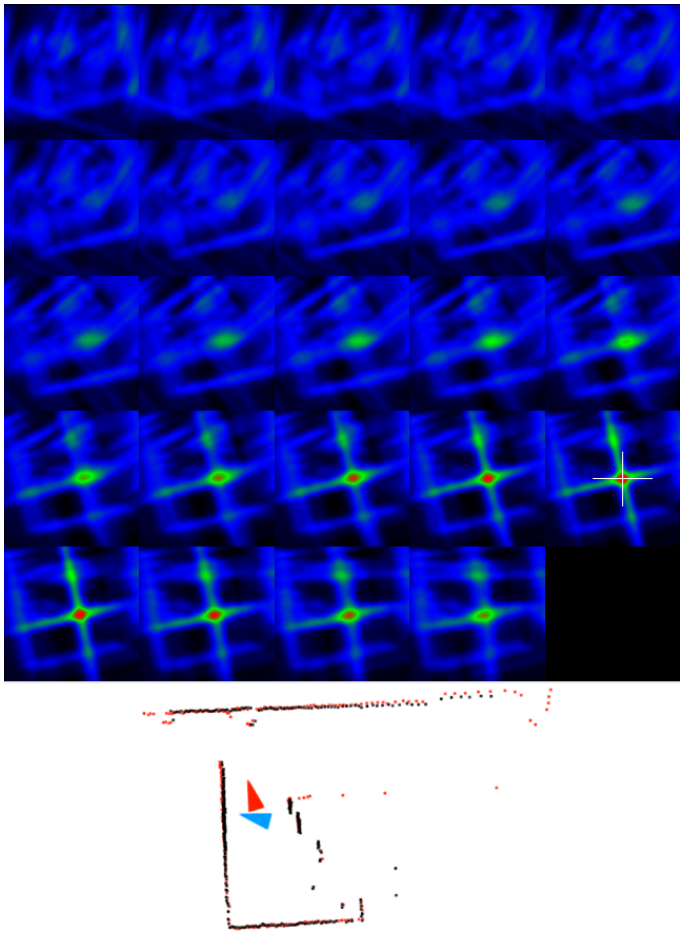
Improvement: "Bad" correspondence pairs are discarded.



Correlative Scan Matching

(Edwin B. Olson)

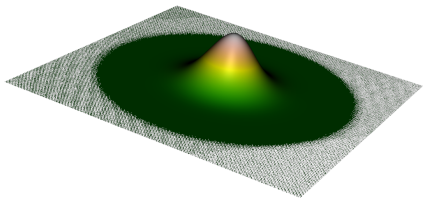
- Shape of a cost function is really complex with number of local extrema.
- Initial position estimation (odometry) can be inaccurate.
- ICP finds a local optimum only.
- Input: two scans
- Goal: find a global optimum

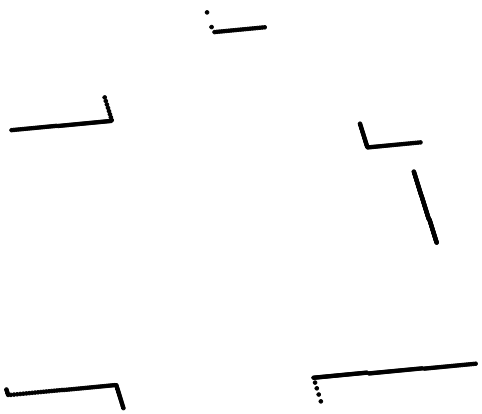


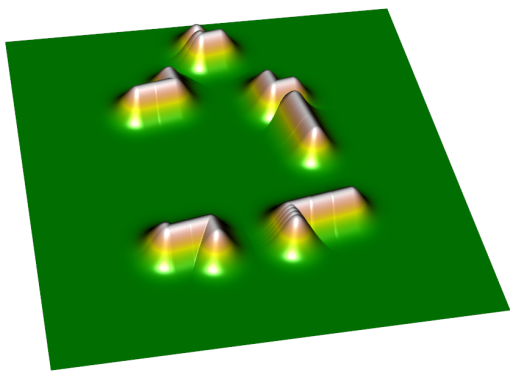
Correlative Scan Matching

Look-table building

- For each point in a reference scan (map) compute the probability that the point corresponds to a reference measurement.







Correlative Scan Matching

Algorithm

- Brute force
  - Compute correlation for each pose (including orientation) of the scan.
  - The pose with the highest correlation is chosen.

```
foreach Δx do
    foreach Δy do
        foreach Δφ do
            foreach point in the current scan do
                compute correlation
            end
        end
    end
end
```

Correlative Scan Matching

Algorithm

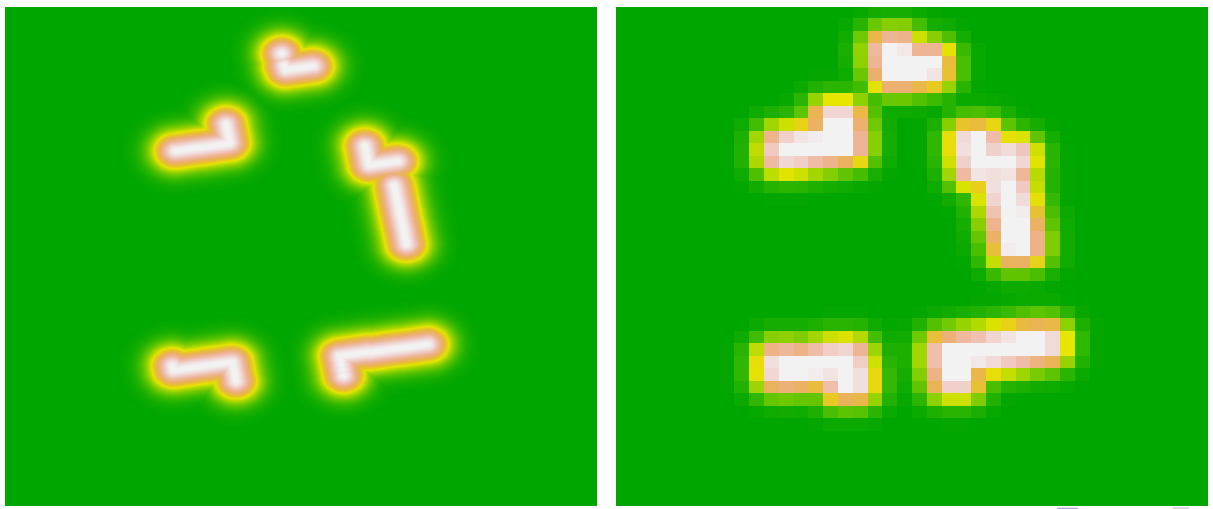
- Computing 2D slices
  - A huge amount of time is spent with scan transformation.
  - Idea: rotation is done in the outer-most loop. The inner loops simply translate the query points.

```
foreach Δφ do
    foreach Δx do
        foreach Δy do
            foreach point in the current scan do
                compute correlation
            end
        end
    end
end
```

Correlative Scan Matching

Algorithm

- Multi-level resolution
  - Two lookup tables: low (LRT) and high resolution table (HRT)
  - Cell in LRT is the maximum of corresponding cells in HRT
  - This guarantees that we will not miss the maximum.
  - Algorithm
    - Compute correspondences in LRT.
    - Find the best voxel V in LRT not yet considered.
    - If the value of $V < H_{best}$ terminate (H_{best} is the maximum).
    - Evaluate V in HRT and set H_{best} if new maximum is found.



Correlative Scan Matching

Results

PC: Intel Core2 6600 @ 2.4 Ghz

	ICP	ICL	Hill-Climb	Correlative (Naive)	Correlative (2D Slices)	Correlative (Multi-Res)	Correlative (GPU 7600GS)	Correlative (GPU GTX260)
0.5 m, 20 deg	56 ms	64 ms	1.1 ms	246 ms	27 ms	8.4 ms	98 ms	26 ms
2.0 m, 40 deg	99 ms	105 ms	1.3 ms	7512 ms	692 ms	20.8 ms	1563 ms	289 ms
4.0 m, 90 deg	145 ms	174 ms	1.0 ms	65282 ms	5029 ms	86.1 ms	13166 ms	2012 ms