Task Sheet 4

Potential field control and collective behaviors

Deadline 15.00, 23.05.2024

Lecture: *Collective Robotics and Scalability*, Summer Term 2024

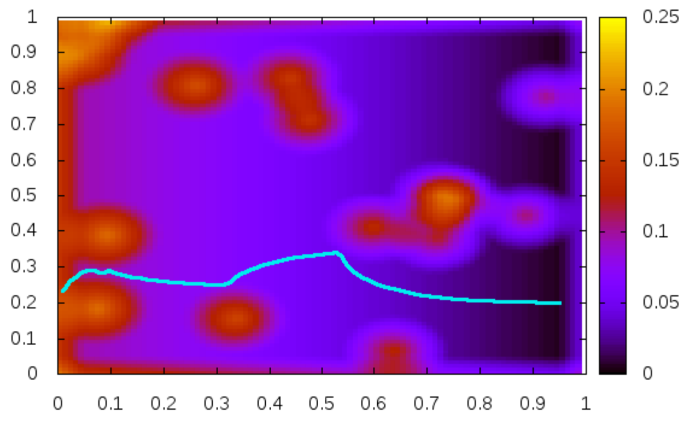
Lecturer: Prof. Dr.-Ing. Heiko Hamann

Tutor: Simay Atasoy, Till Aust

Objectives:

- ▷ investigate limitations of potential field control
- ▷ learn how to program swarm behaviors

Task 4.1 Potential field control



Implement a data structure for a potential field P (e.g., an array). Generate a potential field with a slope in one direction over the whole length (in the above diagram: high potential on the left, low on the right), slopes leading away from the borders, and add a dozen of obstacles in the form of local maxima within the potential field (e.g., cone-shaped, Gaussian – as you wish). Place the robot randomly at a position with high potential at one side (in the above diagram: robot is started on the left).

a) Generate trajectories for several potentials and robot placements with a controller that reacts **directly** to the gradient of the potential field, that is, the slope in x- and y-direction of the potential field at the robot's current position directly determines the robot's **velocity** v. The robot's displacement in one integration step is

$$\Delta x(t) = v_x(t)\Delta t,$$
$$\Delta y(t) = v_y(t)\Delta t,$$

and the velocity is determined by the gradient of the potential field:

$$v_x(t) = \frac{\partial}{\partial x}P(x,y),$$
$$v_y(t) = \frac{\partial}{\partial y}P(x,y).$$

An approximation of the gradient is obtained easily by subtracting two neighboring values of the potential field in your data structure $P[\cdot,\cdot]$:

$$\frac{\partial}{\partial x}P(x,y) \approx P[i,j] - P[i+1,j],$$

for appropriate indices i and j. You have to play around with parameters to make it work: choose an appropriate integration step Δt and/or nominal velocity v, optionally normalize the slope of the potential field to one and scale it by a factor up or down etc.

b) Generate trajectories for several potentials and robot placements with a controller that reacts **indirectly** to the gradient of the potential field, that is, the slope in x- and y-direction of the potential field at the robot's current position only determines the **acceleration** of the robot. The robot's displacement is still determined by

$$\Delta x(t) = v_x(t)\Delta t,$$
$$\Delta y(t) = v_y(t)\Delta t,$$

but the robot's velocity is now determined by

$$\Delta v_x(t) = cv_x(t-\Delta t) + \frac{\partial}{\partial x}P(x,y),$$
$$\Delta v_y(t) = cv_y(t-\Delta t) + \frac{\partial}{\partial y}P(x,y),$$

for $0 < c \ll 1$ and followed by a normalization $\mathbf{v}/|\mathbf{v}| = (v_x, v_y)/|(v_x, v_y)|$ to prevent a too big increase of the velocity. That way the robot can have a kind of memory of former directions. Experiment with different parameters for the steepness of obstacles, normalize the acceleration vector, and implement a discount factor c that reduces the influence of former accelerations. What is a good setting such that the robot always (or at least most of the time) reaches the low values of the potential?

Task 4.2 Behaviors of robot swarms

In this task we focus on behaviors of robot swarms. Initialize your robot arena with 20 randomly distributed robots. All robots should be operated by instances of the same controller (i.e., copies of the same controllers).

a) Program a behavior that makes robots stop if they get close to another robot. For that your simulated robots require to have sensors that detect close-by robots.

b) Extend your program to limit the time a robot stays stopped according to a defined waiting time. When a robot wakes up and moves again it might immediately stop again depending on your implementation. However, the idea is that the robot leaves at least small clusters before stopping again. Think of a strategy to implement this behavior and change your program accordingly.

c) Tweak your implemented behavior and parameters such that your robot swarm eventually aggregates in one big cluster with robots leaving only from time to time but rejoining it soon again.