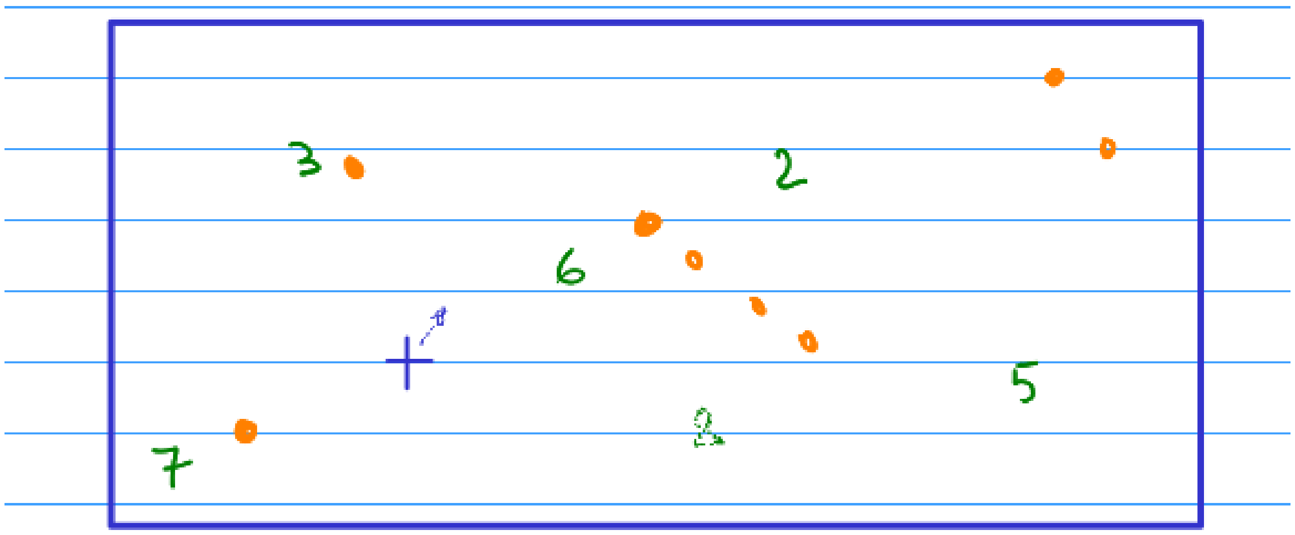
ARP 1st semester assignments V3.0

The project consists in a drone operation interactive simulator.

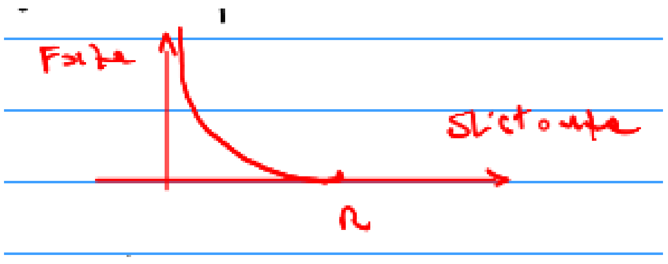


A full screen character window (except one small lateral inspection window). Use **ncurses**.
Blue cross: the drone
Green numbers: targets to reach in sequence – they appear randomly and disappear after reaching
Orange dots: obstacles – they appear randomly and randomly disappear

The drone is operated by keys of the keyboard: 8 directions, plus keys for stopping, resetting, suspending, quitting…

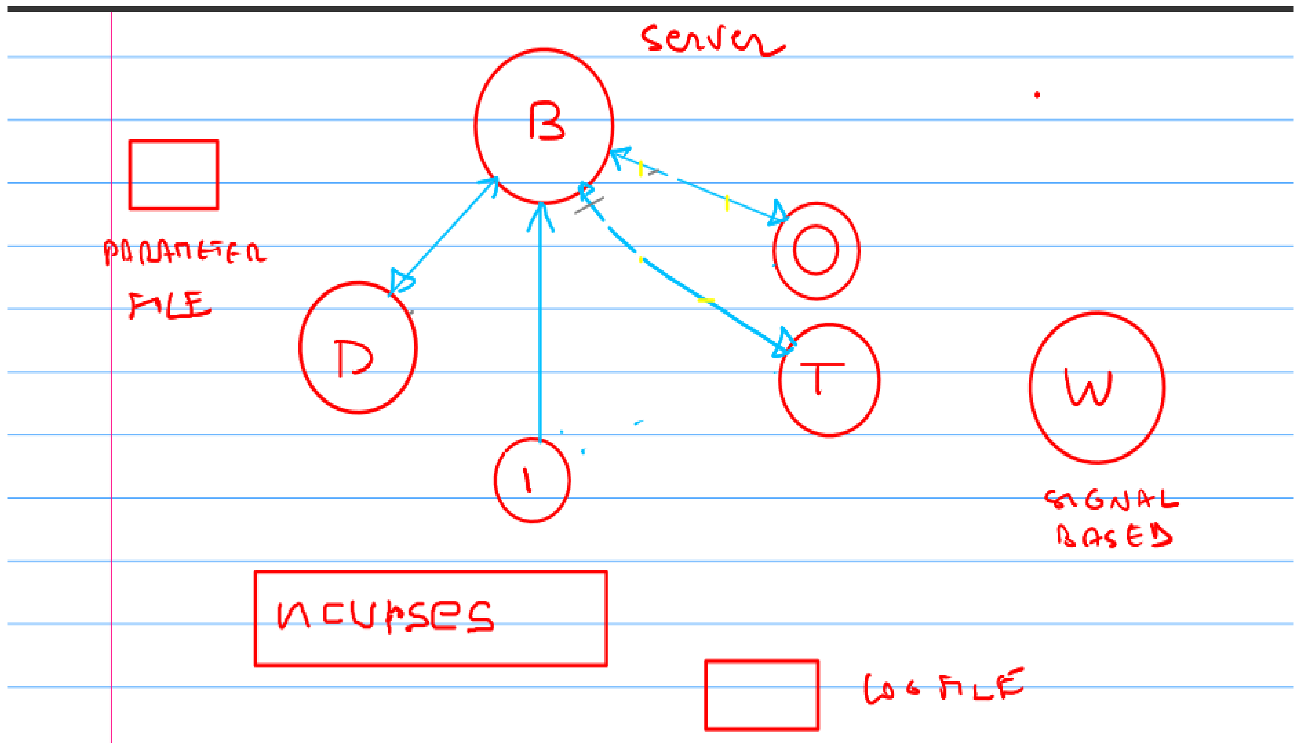
The drone dynamics is a 2- degrees of freedom dot with mass (inertia) and viscous resistance. Any key pressed increases (decreases if reversed) in steps a force pushing the drone in the appropriate direction.

Obstacles repulse the drone. The repulsive force uses the simplest Kathib's model:



in which the repulsive force depends on the distance between the drone and the obstacle, is directed along the shortest line, has a limit radius of perception, and diverges when the distance goes to zero.

The sides of the operation window are obstacles as well. *They simulate the geo-fences used in drone operations.*



The software architecture must have at least 6 active components, plus the window(s), a parameter file and and a logfile.
A possible (but emendable) structure is depicted above. The server manages a ***blackboard*** with the geometrical state of the world (map, drone, targets, obstacles…). Other processes generate the targets and the obstacles. Communications are via pipes and/or signals. All parameters shall be in a file and can be changed in real time). A small inspection window shall show what is happening during execution; its values can be redirected into a logfile. A Watchdog process sends a notification if no computation is going on (must be decided) and successively stops the system.

An overall score shall be computed (with a weighted formula counting the time, how many targets, how many obstacle, distance traveled, possible penalties and so on).

The windows is implemented with the ncurses library. You may choose any key for motion control. Suggested:a cluster of 9 keys like

```
w e r          u i o
s d f    or    j k l       or arrows
x c v          n m ,
```

The center key can be associated to the brake command.
Other keys: can be associated to start, reset or whatever else.

Do not use the mouse.

B: Blackboard server; D: drone dynamics; I: keyboard manager; O,T: obstacles / targets generators; W: watchdog. Each process is responsible of its log data.

Dynamics

The Drone has two degrees of freedom (dofs) and a zero radius body. However, it has a mass (inertia) and a viscous friction due to the air.

External forces are applied to the drone, which moves according to the law of dynamics. They may be:

1. command forces, generated by pressing the keys
2. repulsion forces, generated by obstacles using the Latombe's model
3. (optional) attractive forces, generated by targets using the same model

The general motion equation of the drone is the following:

(1) $\sum \boldsymbol{F} = M \frac{d^2 \boldsymbol{p}}{dt^2} + K \frac{d \boldsymbol{p}}{dt}$

where:
p drone position
F sum of all forces (1, 2 and 3 above) [N]
M mass [Kg]
K viscous coefficient [N· s · m]

Suggested initial values (may change during test):

|**F**|: in steps of 1 N (each pressed key)
M: 1.0 Kg
K: 1.0 N· s · m
Working area (geofence): 100 m width

Digital solution of the dynamic equation

The equation (1) is written as a couple of scalar equations, for the X and Y components. Let us consider for example the only X component:

(2) $\sum F_x = M \frac{d^2 x}{dt^2} + K \frac{d x}{dt}$

The equation (2) can be solved numerically by the Euler's method:

(3) $\sum F_{x_i} = M \frac{x_{i-2} + x_i - 2 x_{i-1}}{T^2} + K \frac{x_i - x_{i-1}}{T}$

where T is the integration interval.

Suggested initial values (may change during test):

T: 10 ÷ 100 ms corresponding to 100 ÷ 10 Hertz simulation cycle.

command (motor) forces

Motion commands correspond to forces in one of the 8 directions. Initially all forces are zero. With each press of a key a constant force is generated incrementally in the direction corresponding to the key. To decrease the force, you push a key representing the opposite direction. Forces are combined using the vector algebra. Command forces are the first component of F in equation (3).

repulsion (obstacle) forces

Obstacles generate repulsive forces, using the Latombe / Kathib's model (see attached images). The two parameters strongly affect the drone's behavior, do that they shall be experimented in trial-and-error tests.

Suggested initial values (may change during test):

ρ: 5 m (remember that beyond this distance the obstacle is not perceived)

η: 0.1 ÷ 10

attractive (target) forces (optional)

Each target may generate a local attractive force, close to the target itself. Use the Latombe / Kathib's model (see attached images).

Preliminary work:

Use ncurses
Build a sample server
Build a watchdog.

assignment 1:

Blackboard Server , Drone, Watchdog (processes B, D, I, O, T)
The server implements the blackboard using a process / pipes / select model
Optionally, it cabe implemented using Posix Shared Memory.
The Window is implemented using ncurses.

Deadline: xxxxxxxx.

assignment 2:

Full system. The I process (keyboard input) is disconnected and substituted by a RPC call model. The process I is thereafter changed. Your program can communicate with other programs in the network.

Deadline: before the exam.