Telescopes

Introduction

Telescopes. Simulate an array of N telescopes (positioned at the bottom of the screen) controlled to point at the centroid of the image of a moving planet (passing on top of the screen). Each telescope introduces some random noise, hence the system integrates the various images to produce an average output image. All images must be shown on the screen. The program must allow the user to change the noise level and motor control parameters of each telescope.

Parameters modification

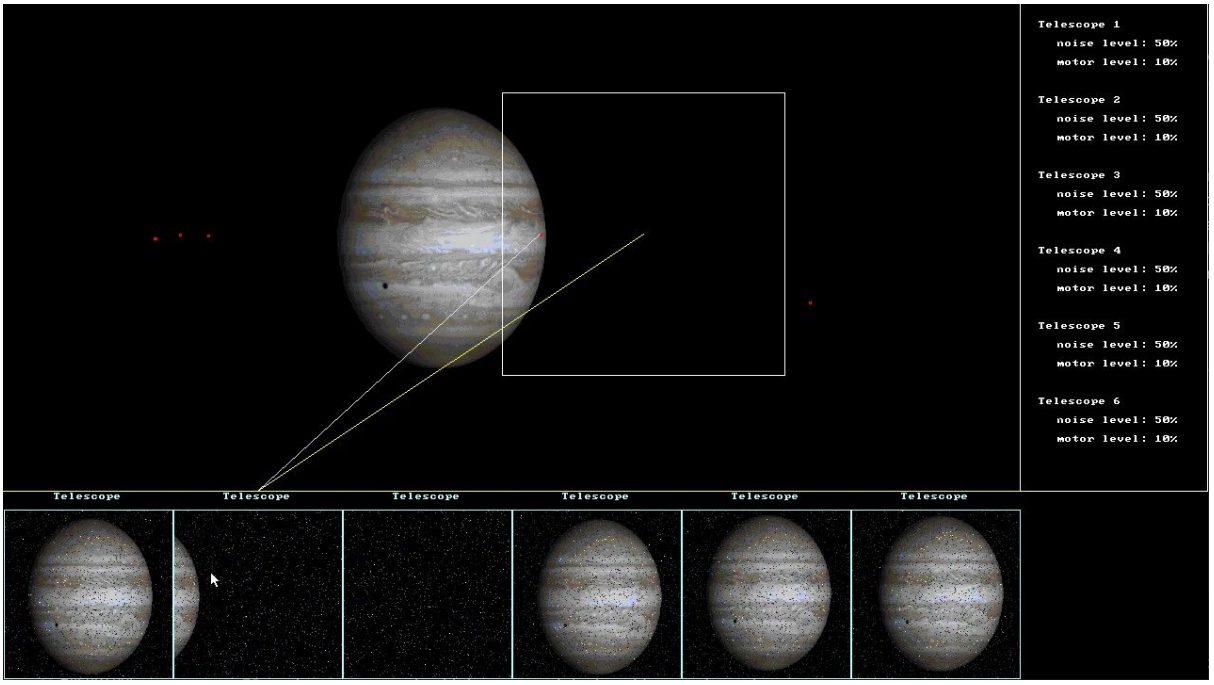
The first interface enables the user to modify noise and motor (velocity) parameters of each telescope. The values should be inserted as a percentage (so a value from 0 to 100), but the application also handles other cases:

- If a numeric value greater than 100 is inserted, the parameter will be set to 100
- If a numeric value lower than 0 (or 10 for motor parameters) is inserted, the parameter will be set to 0 (or 10 for motor parameters)
- If no numeric value is inserted, default parameters[1] will be used
- If a field is left blank, default parameters will be used

[1] 50% for noise and 10% for motor parameters

Simulation environment

Once the user selects the “OK” button of the parameters modification interface, the next phase starts.
The simulation consists in 6 telescopes, each with its observation window positioned randomly in the sky, waiting for the planet to pass in their area. Once the planet enters an observation window, the telescope starts to compute the predicted centroid of the moving object and moves its motor to center it. In this way, the telescope moves in the direction of planet’s centroid and, when it’s reached, takes a picture and stops.
When all telescopes got an image of the planet, the computation starts and outputs the result image.



This interface presents three zones:

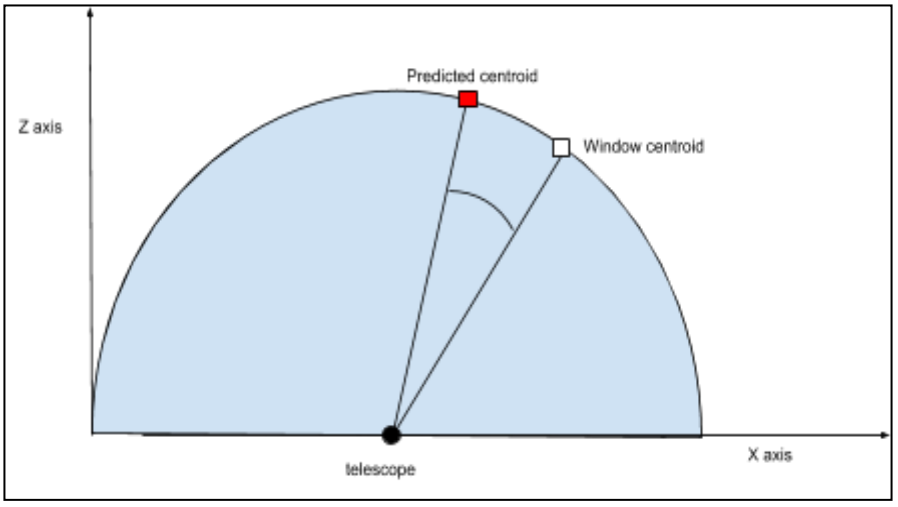
- **Monitor zone**. At the bottom there are six squares, one for each telescope. The images inside are what a telescope is seeing in that moment. When planet’s centroid is reached, the picture (taken by the telescope) is displayed.
  If the user moves the mouse on one of the monitors, the observation window of the relative telescope is displayed in the sky zone
- **Sky zone**. It is located on the top of the monitor zone and displays the moving planet and the observations windows. The red squares are the predicted planet centroid of each telescope
- **Information zone**. Situated on the right side, this zone delivers informations about the parameters (set in the start interface) of each telescope

When the result image is computed, the user is invited to press the “ESC” button to close the program.

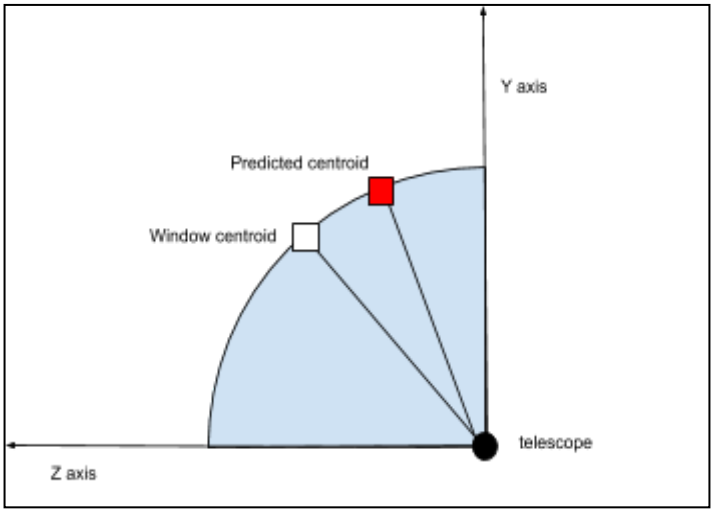
Telescope motion

Observation window motion is based on the angles computed between the predicted centroid and the current center of the window. Those points must be considered on an arc of circumference that has its center on the location of the telescope. Two different cases must be considered:

- X axis angle:



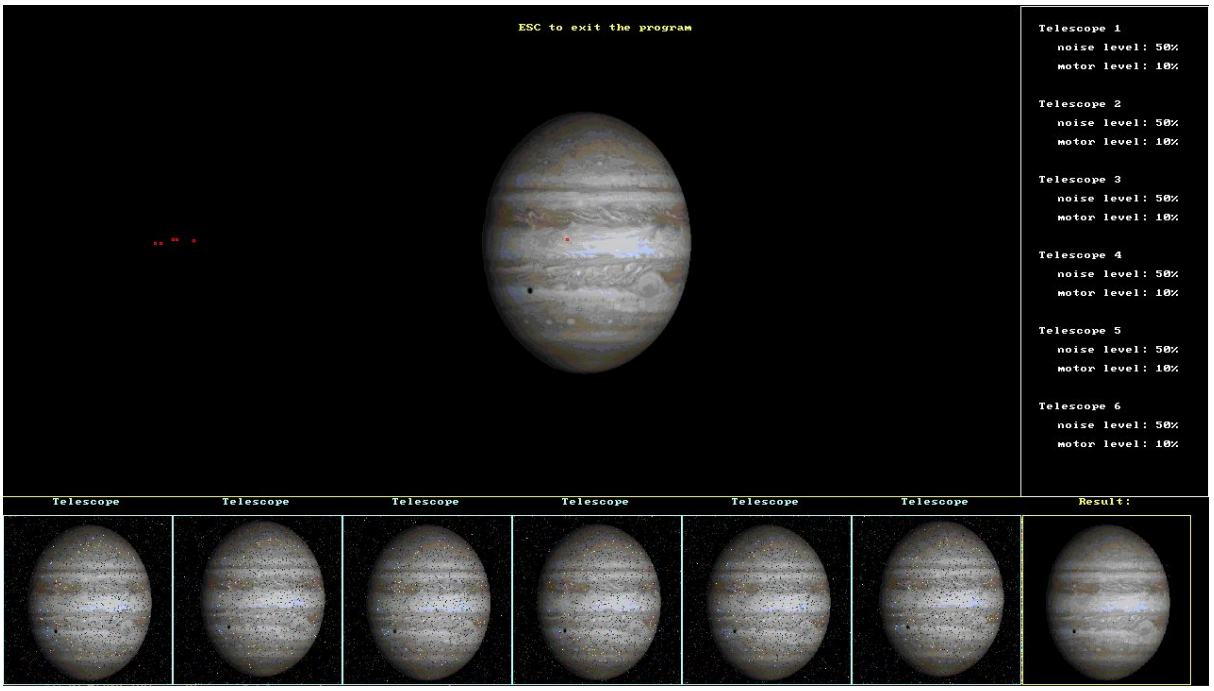
- Y axis angle:



Two angles are then computed and sent as inputs to two motors that move the observation window on the respective axis.

The velocity depends on the parameter set by the user and on how close the window is to the predicted centroid: the closer it is, the less it is fast.

Result image



The final image is computed using the pictures taken by the telescopes, each with its random noise. The result image is shown at the right side of the monitor zone and it is clearly better (less noise) than the original ones.

Code structure

The code is divided in 3 source files:

- main.c; creates all the threads
- observatory.c; contains:
    - Initialization functions
    - Planet update function
    - Telescopes acquisition functions
    - Telescopes motor functions
    - Result image compute function
- gui.c; contains all the graphics related functions of the simulation

and 2 header files:

- observatory.h; contains all the definitions, the structures and the global variables
- gui.h

Task synchronization

There are these tasks:

- A task for the planet motion
- 6 tasks for telescopes image acquisition
- 6 tasks for telescopes motor update
- A task for the final computation
- A task for the gui

To synchronize those tasks there is:

- A mutex for mutual exclusion
- A semaphore for the final computation task

The simulation starts all the threads, but the final computation one is blocked until all images are acquired.
Telescopes can be in various states:

- Observation. In this state the telescope observes the sky waiting for a planet to pass
- Tracking. In this state the telescope computes the predicted centroid each period and moves the observation window accordingly
- Acquired. Planet's centroid has been reached, the telescope takes a picture and stops motion. Final computation task is activated by acquisition task (then motor and acquisition tasks ends)
- Completed. Result image has been computed (computation task ends) and the user is invited to exit the program