Boat and sea wave Blender simulator

Manuel Cortés Batet batetcortesm@gmail.com
ENSTA ParisTech

March 2019

1 Project Abstract

Improving ship stabilization in difficult sea state will require observing incoming waves to anticipate their effect. With this goal in mind, we address the problem of predicting a ship's motion, i.e., pitch and roll angle from sea wave videos. We built a Blender simulation to create a dataset and compare several neural network architectures that, given an image sequence, are trained to predict the future pitch and roll angles of the boat. Current work aims to create a tool to provide the dataset that it is needed.

2 Background

Ship motion is defined by the six degrees of freedom that a ship can experience, which are pitch, roll, yaw, heave, sway and surge. We mainly focus on pitch and roll as prediction targets as a first step, because yaw is mainly controlled by man (fixed in our case), and heave, sway and surge are more difficult to measure, and relatively less important.

3 Main characteristics of the simulator

We built a simulator using Python and Blender that generates 96x54 pixels images seen from a stabilized camera placed at the top of a boat, and simultaneously logs, according to different sea currents, the boat pitch and roll angles. Data augmentation is performed by changing image saturation and brightness. The boat is floating (i.e., not navigating, in the initial simple setting) according to a simple simulation as a cube aligned with the sea surface. The simulator allows to configure several parameters (wave choppiness, wind scale, wind velocity and wind-wave alignment, sea depth -influences the height of the wave-, camera height and angle in the boat, image frequency and resolution)

In the sequence below, several and different characteristics are shown. From left to right the amplitude of the waves and the choppiness have been increased.

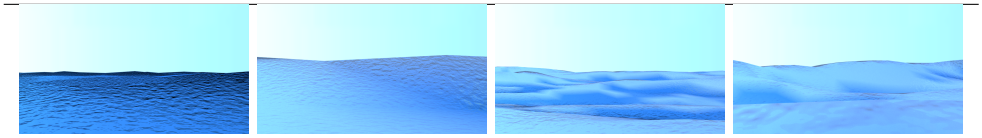

Figure 1: Varied sample images from our simulator

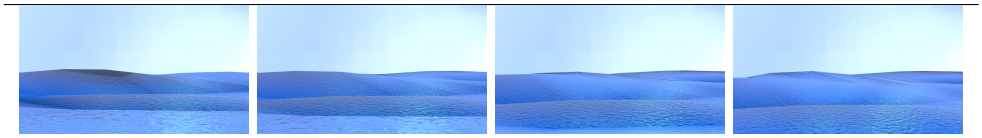

Figure 2: Sample sequence from our simulator

The last sequence is temporal. From left to right, the time of the simulation is increased. It has to be highligth that they are not successive images because, for the human eye, it would be very difficult to notify the difference. Instead of successive images, images with 10 samples of difference have been chosen.
This are images directly used for the training so, the definition is not very high, setled as a engaged between definition and computation time.

4 User Instructions

The instructions for directly using the program are summarized here:
- Install bpy package for Python (Blender as a python module), available at pip application (just do pip install bpy). Documentation if needed: https://pypi.org/project/bpy/
- Download all the files from git: https://github.com/manubatet/Ship-simulator.
-It has to be taken into account that it is not directly executed by python, it is executed by Blender. So open the terminal and go to the 4macro.py file.
-Create a folder named 'data' (in the same path where 4macro.py is) where the different episodes will be saved.
-Once there you have to know the location of blender's app. It is usually in your installation folder and it depends on where did you installed it. You have to execute the blender's app with the terminal. Blender execute python also, so an example for the execution will be:

/Applications/Blender/blender.app/Contents/MacOS/blender -b -P 4macro.py

where -b means background execution and -P means python language

Please note that blender.app/Contents/MacOS/blender is for mac's case, usually its enough to refer the application (do not confuse it with the application folder)

5 Sources

In this section I am going to indicate the main tutorials I have followed to achieve the simulator. Besides the sites I am going to indicate I have consulted a huge amount of forums.
As Zhi (the precedent internship) recommended I have followed this youtube tutorials to create the trajectory and make the ship follow the sea's movement: `https://www.youtube.com/watch?v=sfi7HW8qHAo`.

And for rendering the sea, after trying very hard to define the propierties for the rendering with a python script and not finding any information about how to do that, I took the next tutorial: `https://blenderartists.org/t/fast-ocean-blender-add-on-for-ocean-foam-and-auto-collisions/688073/44`

6 Algorithm for the simulator

Due to the impossibility of putting into a python script the render part, it is mandatory do use a blender file with the propierties of the render already set. The macro opens this blender file, modifies the physical parameters of the ocean, introduces the boat and the camera and takes the data (frames and pith and roll).

As i said before, the trajectory of the boat is a circle. The first thing the algorithm does is to create a plane and subdivide it twice. After that it makes the plane follow the circle but with the condition that the vertex of the subdivision should be together and setting the velocity of the plane. After that it creates a cube and it links it to the plane computing the middle between the vertex named before and the center of the cube.
After that it creates a camera and links it in a constant size from the cube to make. The link is also for the angles of the camera with respect to the cube but the pitch and the roll of the camera are disabled so it will be stabilized no matter the pitch and the roll of the ship

7 Parameters that are not in the script

There are some parameter that are not in the script but can be easily change in order to have more and different data

7.1 Change the sky

Changing the sky will not change only the sky, it will change also how the light affect the water and how it is reflected. It is very easy to change. Just open blender and go to the 'world' option in the Properties (normaly at the right part of the screen) and go to surface. There will be an image .jpg or .png selected,

go to the folder draw and select the new one. Take into account that you will need a 360 degrees sky image, not just a photo.

7.2 Link again the camera and the boat

To link the camera and the boat go to the camera's properties and to the constrains options (chain drawing) and select 'X' and 'Y' in 'Rotation'.

7.3 Changing the characteristics of the boat

We have thought about changing the characteristics of the boat in a future. As it is explained before, the boat is linked to a plane so if we want to simulate a bigger boat where the area of the influence between the sea and the boat is bigger we will just have to do the plane bigger. With this, somehow we will simulate the inertia of the boat that is not taked into account in the simulator (there is no any physical model in the simulation)

7.4 Changing the render properties of the sea

The level of realism of the sea can be improved changing the properties of the sea if it is need, you can follow a tutorial about that in the internet and change the blender file (oceanrender2.blend) and save it. I tried several and i thought this one was fine in order to do not take a lot of time computing each frame.

8 Files

8.1 parameters.py

Here you have all the parameters that the macro is goingo to import. You can select to iterate between several parameters. Those parameters are only choppiness, wave scale, wind velocity and random seed. Is not very difficult to add more parameters but those are the parameters that are more important to change the simulation. Here: https://docs.blender.org/manual/en/dev/modeling/modifiers/simulate/ocean.html you can find what does each parameter exactly do related to the ocean creation. The parameters about the render are clearly explained in the code.

8.2 macro.py

This is the macro. Theorically you will not need to change anything from here but there are some explanations inside about what the code does and I encourage you to improve the code and add more functions.

8.3 VideoMaker.py

This is the program to make videos of the different episodes. You have to specify the folder of the episode and the folder where you want to save the video.

Example (done in the same path as data folder is):
python3 VideoMaker.py -i data/2021204820 -o data/20190421204820