Title **Efficient, Safe and Sustainable Traffic at Sea**
Acronym **EfficienSea**

Contract No. 013

Document No. W_WP5_2
Document Access: Restricted

A numerical navigator

Date: 23.01.2012

Part-financed by the European Union (European Regional Development Fund and European Neighbourhood and Partnership Instrument)

DOCUMENT STATUS

Authors

Name	*Organisation*
Erik Ravn	DAMSA

Reviewing/Approval of report

Name	*Organisation*	*Signature*	*Date*

Document History

Revision	*Date*	*Organisation*	*Initials*	*Revised pages*	*Short description of changes*
1. draft	23-01-2012	DAMSA	esr	7	

Part-financed by the European Union

Content

Part-financed by the European Union

1 Scope

This report describes how a numerical navigator has been modeled. Hydrodynamic forces have not been included. The ship has a given turn radius. Only collision has been implemented, not grounds. The navigator is rule based. If another ship is on a collision course it will turn to starboard. It is given an initial set of waypoints which it will attempt to steer on. If it during its voyage has to avoid another ship then its waypoints are recalculated. For instance if the ship is forced 500 m. to the right, then it might not necessarily have to steer 500 m. to the left, but might be able to continue parallel to the defined leg and then turn at another waypoint. A route is defined by a set of legs. To test the navigator ships are fed into the beginning and end of the routes using a poison process and the given traffic distributions of the leg. A Google Earth file showing the results is generated by the developed software.

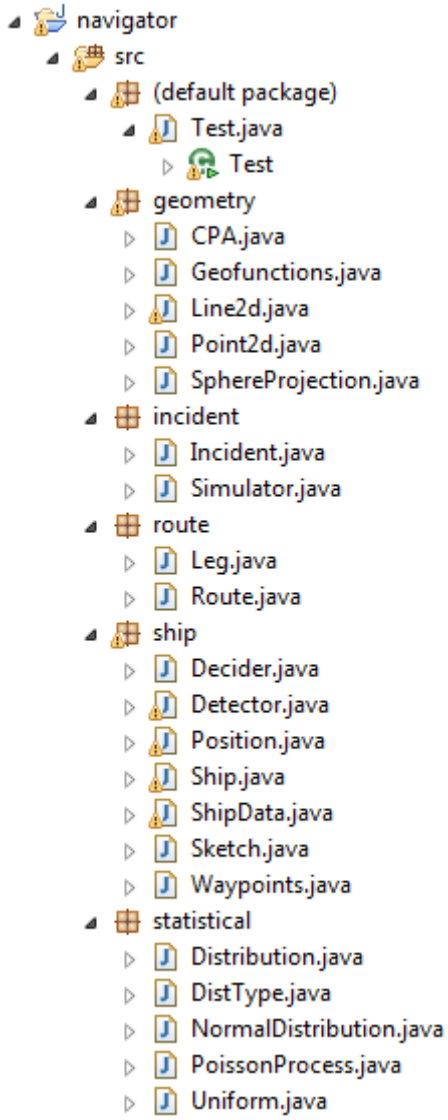
2 Background

The reason why a numerical navigator could be useful is for simulating the traffic in the waterways. In this way hotspots and congestion areas could be identified. This could be used when waterways are relocated or the traffic is thought to increase.

Part-financed by the European Union

3 Overview

The program can be found together with this document. It is coded in java and has the following structure:



Test.java
This is the function where the program starts
First the waypoints, legs and routes are defined.
Then an object called Simulator is defined and the routes are added to this.
The method simulate is called.

Waypoint.java
This class creates the waypoints that the ship should pass through in order to complete its journey. From the legs of the route, the lateral distributions and the turn radius of the ship a waypoints are generated. These waypoints can be updated during the journey if the ship has to make an evasive maneuver in order not to collide with another ship.

Part-financed by the European Union

Simulator.java
A loop running from t=0 to t=x is now entered.
Here we add new ships to the routes and remove ships that have reached their destination.

Another Simulator object, simulator2, is defined. This is a copy of the previous defined simulator, but this is always in front of the first simulator by t seconds. Every minute simulator2 will look 5 minutes forward and extract the ships that it will collide with.
Now a loop runs through though each ship and determines where it is, if there are other ships are of interest and what the ship should do.

Detector.java
This class is used by the ship to look around and detect other ships
The main function is called scan(…) and calls the other functions in the class.

Decider.java
This class uses the results from Detector to make decisions on what the ship shall do. The main function is called makeDecisions(…).

Part-financed by the European Union

4 Testing the navigator

A test scenario has been created with two routes. Ships of random size are added to the routes using a poison distribution. When running this a large number of kml files are generated. In order to integrate them into one single kml file open the Excel file 'OpenKMLs.xlm' and look in the vba code.

Initial data:
Route as below.
On both route the ships are distributed 300m to the right with zero standard deviation. This ensures that we have ships on collision course.
The ships are added according to a poison distribution with intensity 50000 ships per year.
A simulation on 6 hours was performed and 1 collision was registered.

Figure 1: Screenshot of the test scenario simulation

Part-financed by the European Union

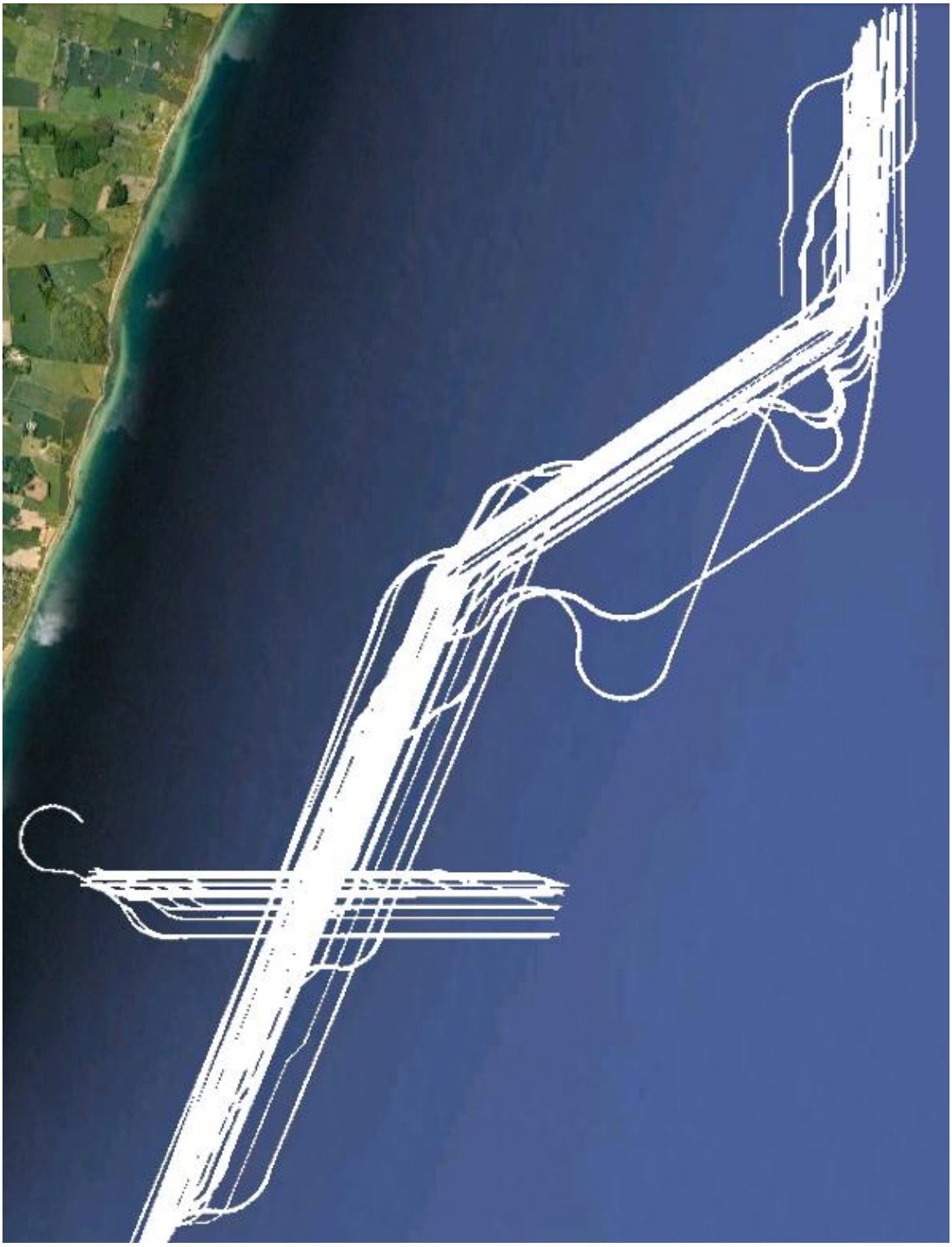


Figure 2: Tracks of the ships in the test scenario simulation

Part-financed by the European Union

5 To Do

Some hydrodynamic properties should be implemented, such as acceleration/deceleration and current/wind.

Include bathymetry/grounding

Include aids to navigation

6 Conclusions

A java program that attempts to simulate the ships along a number of routes has been created. On collision has been implemented. Hydrodynamics aspects of the ship are not considered. When testing the navigator it avoids ships on collision course.

Part-financed by the European Union