Planning a route for an unmanned autonomous vessel

Emilian Świtalski *

* *Gdynia Maritime University, Poland*

Abstract. The work discusses the problem of optimal route planning for autonomous vessel in order to depth measurements during repositioning. This data can be used to create bathymetric map. The paper describes the application, which allows determining the measurement route in a quick and efficient way. The application was created using web technologies: PHP, JS, Google Maps Api.

Key Words: anchor-trail, autonomous, scheduling, routing, route, trail, coordinates, bathymetric measurements

Introduction

In 2016, along with other students of the Gdynia Maritime University, I have modernized a small remote-controlled vessel into autonomous mode. The purpose of this vehicle was to provide data for the preparation of bathymetric maps, that is the depth as a function of the position. Measurement has been done with a single beam sonar [5].

Although completing this task seemed to bring positive effects, in order to achieve optimal results, the whole project would have to be redone. Because presented issue focuses on many areas, from mechanics through automation, electronics, computer science to geodesy, doing it from scratch with proper diligence is currently out of my reach. That's why I've made a decision to deal only with the fragment that was solved in the least optimal way, i.e. the designation of checkpoints on the route.

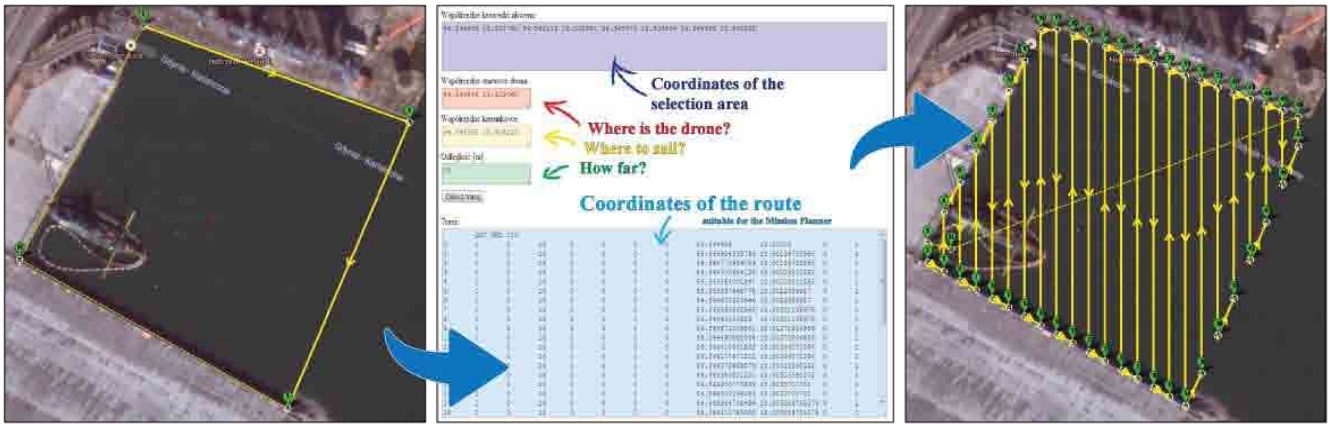

Figure 1. Previous solution (outline, route calculation form, result trail)

Markers were created in the designated area, forming horizontal measurement profiles. Although, skipping the vessel's turning radius when setting the route seemed to be a major omission. The designated route forced the vessel to take very sharp turns that it was unable to perform. As a result, the actual route was not coincident with the planned and vessel's maneuvering could lead to collision.

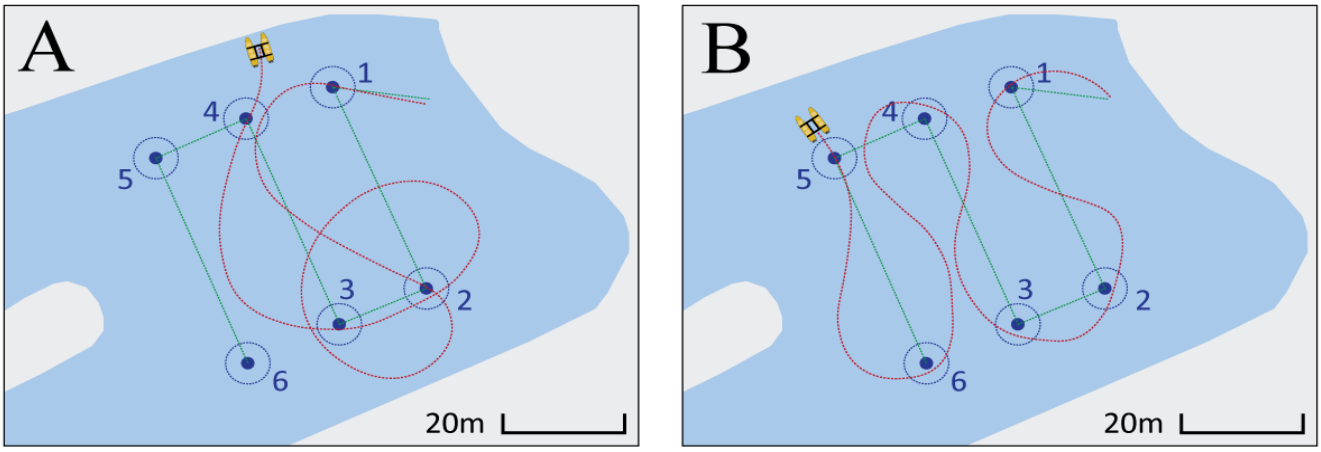

Figure 2. Possible scenarios: A - Greater turn radius (collision), B – Smaller turn radius

During the stand-alone measurements, the use of measurement profiles is not required. However, it is important that the vessel reaches every place from which it is supposed to provide measurement.

This short article will focus on describing the application from the users' point of view, so that those who are potentially interested in the program, after reading the whole would have no doubt whether it meets their requirements and would be able to use it without issues.

Guidelines and Technology Review

Because of the fact that constructing an autonomous water-moving unit is not currently planned by me, the application will be written in such a way it can serve as a ready-made tool for anyone willing to construct such unit.

In that case, the ideal solution from technological point of view will be the use of web technologies such as HTML, CSS, **JS**, **PHP**. So, the presented application is a standard web site that will be available on a given domain: **anchor-trail.eu**.

HTML serves only to create a page structure, and CSS will be used to describe the presentation form of a program. The JavaScript code is executed on user's computer, so that it will be able to handle the functions of the program that need to work instantaneously, i.e. all the planning, data entry and validation activities. Once the user has determined that the route is already planned, all the necessary data can be sent with use of a single button to the server, where the route will be generated by the PHP code. The generation of the route on an external server will provide safety as it will exclude the risk of unauthorized access to a code.

It is worth mentioning that by using JS there is a possibility for a free of charge use of GOOGLE maps. The **GOOGLE MAPS API**, besides the maps, provides users with many useful tools that can be implemented while designing a user interface (GUI). These include markers, polygons, polylines that can be freely placed on the map by referring to the geographical coordinates. All of these elements can be freely grouped and given special features. In a route planning application, each marker group will be creating an area marked with a polygon, and groups will be distinguished from each other by their color. Each of these items can trigger selected events that are to be handled. Events include all the actions performed by the user, such as clicking, hovering over the selected object et cetera. [1][2][3][4]

Graphical User Interface

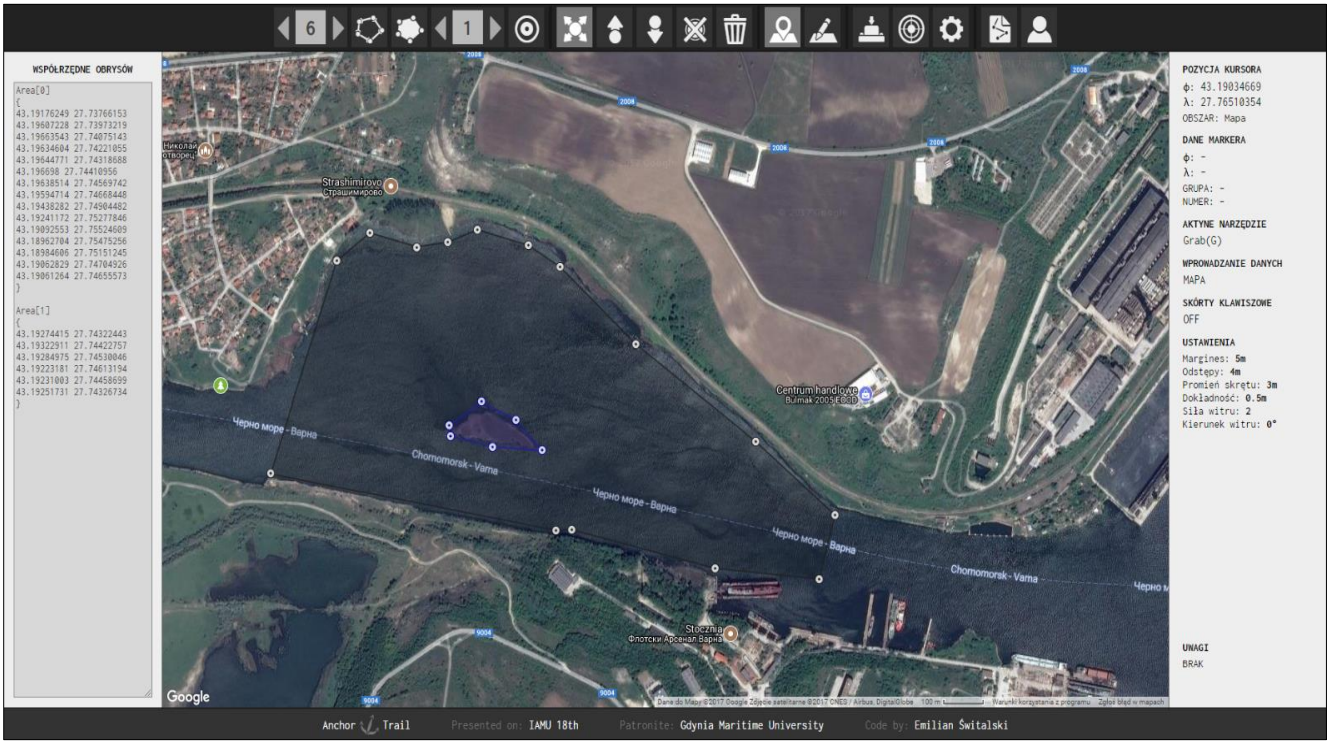

Figure 3. Print screen from the application window

The application is designed to be effective and convenient to use for people who have already used it before, but also approachable for people who are about to use it for the first time.

The application fills up all available browser window space and consists of the following parts:

- toolbar (top);
- coordinate bar (left);
- information bar (right);
- map (center);
- footer (down).

The information bar is to contain all the information that may be useful for the user at given moment. When moving the cursor around the map, its coordinates are being shown, and when pointing the designated area, the bar shows the area's number. In addition, when the cursor is hovering over a marker, the information bar shows all the marker's parameters. The information bar also contains all current program settings. After hovering over any tool, its detailed description will be displayed in the information bar, which can be useful, especially for all the first time users. At the very bottom of the info bar, comments coming from validating data that have not gone through positive, informing about a specific error, will be displayed.

The **Map(M)** and **Text(T)** buttons are important ones on the toolbar. They change the way data is being input into the memory of the program. The Map(M) button, when displayed as being pressed, informs us that the markers on the map can be manipulated and the data in the coordinate bar is being updated on a regular basis. After pressing the Text (T) button, the map will be locked and the possibility to edit the text file, that is the result set of data, will appear. Re-switching to the map will take into account the changes that were previously made in the text file. In some cases, the user will also be informed about using incorrect format during data insertion.

All buttons on the upper toolbar have assigned keyboard shortcuts, that can be activated or deactivated using the **KeyShortcuts(K)** switch. The letter appearing in brackets next to the tool name informs about its assigned shortcut key.

The tool by which the area of the route can be determined is **Area (A)**. When it is active, each click on the map will add another contour marker. Those markers will create the area on which the route will be mapped. In the program, the route stroke markers have a **group of 0**.

In order to bypass any possible obstacles occurring within the planned route, the **ExtraArea(E)** tool should be used. When it is active, each click on the map will add a pair of pointers placed inside the route area to create a terrain that will be bypassed during route generation. The digit placed in the cell to the right of this tool points to a group of added markers of additional areas (obstacles). Additional areas may belong to **groups 1 – 9**.

In order to indicate the starting point from which the vessel will start the route, using the **FirstPoint(F)** tool is needed. The marker added this way will be assigned to **group 10**.

The cell placed to the left of the toolbar contains a number determining the marker number that will be currently added. Markers are numbered with consecutive integers, starting from 0. When the marker number in a cell is smaller than the number of markers in a group, next markers will be shifted.

Markers can be switched with **Up(U)** and **Down(D)** tools. When the Up (U) button is active, clicking on the selected marker (the vertex of the stroke) increases its number at the expense of the next marker, which in practice results in switching places by two chosen markers. Double-clicking will result in setting the marker number to the maximum and adjusting other markers' numbers. The Down (D) tool works similarly, with the exception that it decreases the number of given marker.

The **Grab(G)** tool comes in handy when there is a need to change the position of any of the markers. Once activated, the selected marker needs to be dragged to the appropriate location. Once it is dropped, the outline and position in the text file will be updated.

Deletion of a marker requires clicking on the selected one with the active **Delete (X)** tool. Once the marker has been deleted, the remaining markers will be renumbered.

Removing all the markers from the selected group calls for activating the possibility of adding markers and clicking the **Remove(R)** button. When none of the tools for adding markers is active, all of them will be deleted.

There is a possibility of centering the map by entering geographic coordinates. This can be done via the form hidden under the **Center(C)** button.

It is important to precisely set parameters hidden in the **Settings(S)** tab before generating a route. Those given parameters are as follows:

- margin [m] - safety distance from edges of drawn contours;
- spacing [m] - the distance from which next measurements will be taken;
- yurning radius [m] - smoothing of corners;
- accuracy [m] - the degree of rounding;
- wind power [%] - increases or decreases the turn radius depending on the direction of the rotor (value to be selected intuitively);
- wind direction [°] – inserted in degrees, angle 0 degrees corresponds to the wind blowing from the south to the north.

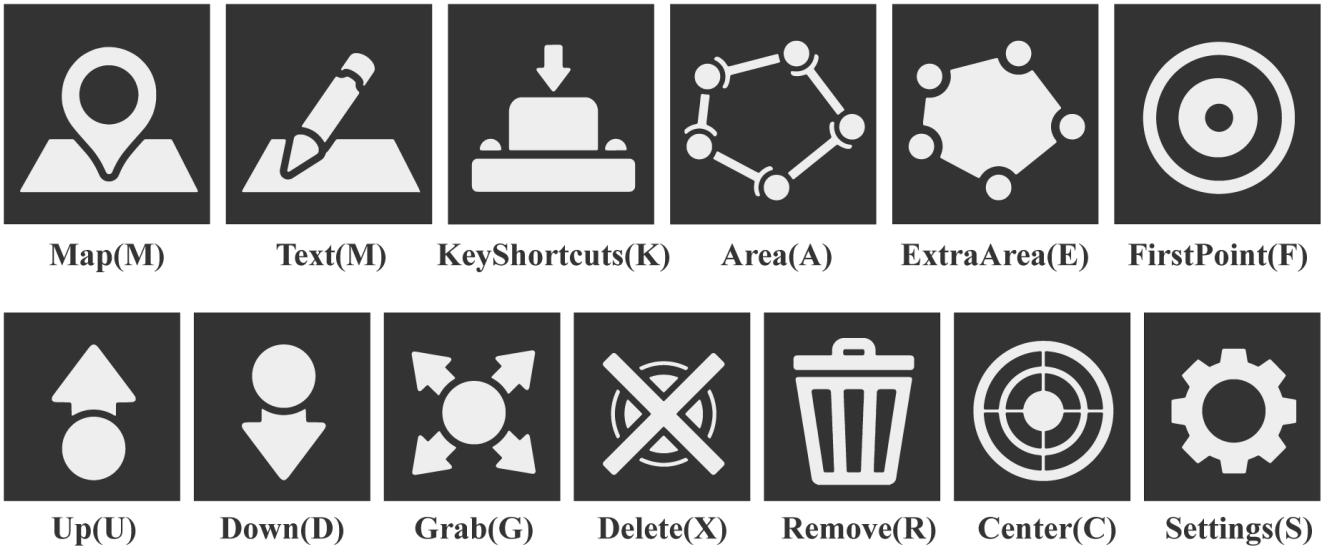

Figure 4. Tool icons along with their names

Generating a trail

When the user assumes that the strokes are set correctly and that all parameters have corresponding values, then the route can be generated using the **GenerateTrail(W)** button. This way all data will be sent to the server. The algorithm placed on the server, using different coordinate systems and basic mathematical operations, transforms the input into a route that is a sequence of consecutive coordinates in the geographic system. Such generated route will be sent back to the user. A new window, designed similar to the route planner, will pop up in the browser. The user then will be able to view the route and make corrections. Once the work is considered to be complete, the possibility to save it in a text file or a special format adapted to the **Mission Planner** autopilot will appear.

The route consists of consecutive points, so all the roundings will be approximated to polygons. Along with decreasing the "accuracy" parameter in meters, the rounding will be mapped with greater precision, but the number of resulting route's points will increase. The density of the measurements, and therefore the accuracy of the future map, is affected by the "spacing" parameter. The smaller it is, the more measurements should be obtained.

GenerateTrail(W)

Figure 5. Button icon for the trail generation

Bibliographic references

1. www.gmapsapi.com/poradnik
2. www.kursjs.pl/kurs
3. www.developers.google.com/maps
4. www.w3schools.com
5. Cezary Specht, Emilian Świtalski, Mariusz Specht. Application of an Autonomous, Unmanned Survey Vessel (ASV/USV) in Bathymetric Measurements