

- 1 -

MARINE SEISMIC SURVEYS

The present invention relates to marine seismic surveys, and in particular for a system and method for generating a route traversing survey lines of a marine seismic survey.

5

Marine seismic surveys are used by oil companies to locate geological traps which may contain oil or gas, and to determine suitable drilling locations. In general, an oil company negotiates with governments to explore particular geographic areas. The company's geologists identify the most promising regions within these areas, and then design survey patterns that they hope will deliver the best information for the lowest cost. Survey patterns can be characterised as "2D" or "3D" survey patterns depending on the coarseness of the pattern in comparison with the features of interest. A typical "2D" survey pattern might consist of one hundred straight survey lines, ranging from 30 to 200 km in length arranged in an overlapping grid; A typical "3D" survey pattern might consist of a similar number of parallel lines more closely spaced.

The survey is carried out by a vessel or ship 101 towing one or more long subsurface streamers 102 and one or more seismic sources 103, as shown in Figure 1. Each source 103 is an array of compressed air guns that generates an intense shock wave or seismic pulse at regular intervals of time. Each streamer 102 is a long cable containing hydrophones along its length. As the ship sails along a straight line, the source 103 produces a seismic pulse that travels through the water and is reflected back towards the surface by structural layers beneath the sea floor. The hydrophones inside each streamer 102 detect these reflected signals, the signal reaching each hydrophone having travelled along a different path. By analysing and correlating the reflected signals detected by the hydrophones, it is possible to construct a profile of the geological structure of the sea bed along the straight line followed by the ship. A map or model of the geological structure of a particular region of the sea floor can be generated from seismic data measured along a series of straight lines forming a mesh over the region of interest. Straight lines are used to facilitate analysis of the data from the array of hydrophones.

30

- 2 -

A typical marine seismic survey pattern 200 comprises a series of straight lines 202, as shown in Figure 2. The ship 101 towing the streamer 102 and seismic source 103 sails along each line, collecting seismic data. The process of travelling from the end of one line to the start of the next is referred to as a line change. During a line change, no seismic data is collected. It is therefore important to reduce the unproductive time spent by the ship 101 on line changes between survey lines. However, planning the order and direction in which to traverse the survey lines in order to improve productivity is not a trivial task. The ship 101 could take any one of an enormous number of possible routes along the series of lines. The problem is exacerbated by the need to ensure that the streamer 102 is straight when commencing to survey a specified survey line, and also by the minimum turn radius that the particular survey vessel can use. Moreover, if the survey area is in a tidal region or in shallow water, it may be important to allow for changing local currents and tide levels over potentially dangerous hazards. Furthermore, the ship's location at a particular time may be important, to meet a helicopter, for example. Furthermore, it may be necessary to allow time to perform maintenance or urgent repairs during line changes.

Today, survey routes are planned manually, resulting in somewhat *ad hoc* routes that are not likely to be optimal. It is desired, therefore, to provide a system and method for generating a route traversing predetermined seismic survey paths in a marine seismic survey, or at least a useful alternative to existing methods and systems for route planning.

In accordance with the present invention there is provided a method for use in generating at least one route traversing a plurality of predetermined seismic survey paths in a marine seismic survey, including:

- 25 generating a plurality of routes traversing at least a subset of said predetermined seismic survey paths, said routes satisfying predetermined constraints;
- generating respective penalties for said routes; and
- selecting at least one of said routes on the basis of said penalties.

30 The present invention also provides a method for use in generating at least one route traversing a plurality of predetermined seismic survey paths in a marine seismic survey, including:

- 3 -

- (a) generating a plurality of partial routes, each partial route including a different subset of said predetermined survey paths, said partial routes satisfying predetermined constraints;
- 5 (b) for each partial route, generating a set of corresponding extended routes by extending the partial route to include respective selected ones of remaining predetermined seismic survey paths, said extended routes satisfying predetermined constraints;
- (c) generating penalties for said extended routes;
- (d) selecting a subset of said extended routes on the basis of said penalties; and
- 10 (e) repeating steps (b)–(d), using said subset of extended routes as said set of partial routes.

The present invention also provides a method for evaluating a seismic survey comprising a plurality of predetermined seismic survey paths in a marine seismic survey, including:

- 15 generating a plurality of routes traversing a subset of said predetermined seismic survey paths, said routes satisfying predetermined constraints;
- generating respective penalties for said routes;
- selecting at least one of said routes on the basis of said penalties;
- generating a route traversing said plurality of predetermined seismic survey paths
- 20 from the selected at least one of said routes; and
- evaluating said survey on the basis of a metric of said route.

The present invention also provides a system for controlling a marine seismic survey, including:

- 25 an interface to a positioning system for determining the current position of a survey vessel; and
- a marine seismic survey system including means for generating navigational data for said survey vessel based on said current position to traverse said route.

30 Preferred embodiments of the present invention are hereinafter described, by way of example only, with reference to the accompanying drawings, wherein:

Figure 1 is a schematic illustration of a seismic survey ship towing a seismic source and a streamer;

- 4 -

Figure 2 is a plan view schematic diagram of a series of predetermined seismic survey lines, together with a survey ship;

Figure 3 is a block diagram of a preferred embodiment of a marine seismic survey planning system;

5 Figure 4 is a schematic illustration of the parameters of a line change path traversed by a survey ship between survey lines;

Figures 5 and 6 are schematic diagrams generated by the survey planning system, showing two routes through the survey lines shown in Figure 2;

Figure 7 is a schematic diagram illustrating feather in a survey; and

10 Figures 8 to 10 are screenshots of displays generated by the survey planning system.

A survey planning system (SPS) 1, as shown in Figure 3, executes a survey planning process that determines the overheads of a large number of possible marine seismic survey routes, based on supplied survey line coordinates and other constraints, and determines the most efficient routes from the set of generated routes; *i.e.*, those traversing the shortest distance or requiring the least time. An implementation of the SPS 1 is described below and includes a standard computer system 2 with disk storage 3 having components provided by software modules stored and executed on the computer system 2, which may be a personal computer running a Microsoft WindowsTM operating system. It will be apparent to those skilled in the art that the components of the SPS 1 may be distributed on one or more different computer systems in different locations connected by a communications network. Also at least some of the steps executed by the software modules of the SPS 1 may be executed by hardware circuits, such as application-specific integrated circuits (ASICs).

15
20
25

A survey pattern is provided to the SPS 1 as a text file containing the coordinates of each survey line in WGS84 or WGS90 (World Geodetic System) format, with one coordinate pair per line. The first coordinate pair in the file is the more Northerly end of the line, but if the line runs purely along an East-West direction, the more Easterly coordinate is given first. Each survey line is also given a name as the first token of the corresponding line of the text file. The SPS 1 reads the text file and generates a display of the survey pattern 200, providing the ability to select individual survey lines 202 for the purpose of

30

- 5 -

specifying line properties, such as assigning priority to a particular survey line. Existing lines may be modified, and new lines may be added to the survey pattern 200. The SPS 1 also provides pull down menus and dialog windows for specifying numerous parameters for generating survey routes traversing the survey pattern 200.

5

The SPS 1 constructs the most efficient line change path to take from the end of one survey line and the beginning of the next, using a number of parameters specified by the user. When the ship 101 reaches the end of a survey line, it continues along the same direction until a specified length of the streamer 102 passes the end of the survey line. This length is referred to as the end-of-line (EOL) run out 401, as shown in Figure 4. Similarly, the ship 101 travels an additional distance along the direction of the next survey line before the start of that line to ensure that a specified length of the streamer 102 is straight before passing the beginning of the survey line. This distance is known as the start-of-line (SOL) run in 402. The path taken by the ship 101 between the end 408 of the EOL run out 401 and the start 410 of the SOL run in 402 is a straight line 403 connected by the curved EOL 405 and SOL 406 turn segments determined by the minimum turn radius 404 of the ship 101. The SPS 1 generates a dialog window that allows the user to specify the total streamer length (bow to tailbuoy), the minimum turn radius 404, the SOL run in 402, and the EOL run out 401. The SPS 1 generates routes traversing a survey pattern that conform to these restrictions.

20

Figures 5 and 6 show two routes generated by the SPS 1 for the set 200 of survey lines 202 shown in Figure 2. In general, runs along two adjacent survey lines are not taken consecutively due to the above restrictions. The most efficient route can also depend upon the current position of the ship 101. Furthermore, routes can be subject to additional constraints, such as priorities or particular directions assigned to selected survey lines, tidal hazards in shallow waters (requiring the ship to avoid those areas at low tides), and the need to allow for maintenance time between selected survey lines. Finally, the orientation of the streamer 102 as it is towed along a survey line may not be the same as the orientation of the survey line due to water currents, resulting in a separation between the end of the streamer 102 and the survey line, known as 'feather', as shown in Figure 7. Feather introduces errors in the survey data because the hydrophones attached to the streamer 102 do not measure data along the desired survey line when feather is present.

30

- 6 -

Therefore, it is desirable to reduce feather to improve the survey quality. It is also important to know the expected feather during surveys in which a number of streamers are towed by a ship. In this case, the feather determines the number of passes required to cover the survey area. This is particularly important at the end of a survey when the 'gaps' in the coverage are 'filled in'. Feather is determined by water currents and tides, introducing additional constraints on the times that survey lines should be surveyed, depending upon their orientations.

The SPS 1 generates quasi-optimal survey routes using a route search procedure. The procedure generates a number of candidate routes that satisfy one or more of the following requirements:

- (i) minimise distance and/or duration;
- (ii) cover priority lines as early as required;
- (iii) avoid tidal hazards at low tides;
- (iv) minimise feather encountered while acquiring data;
- (v) allow required maintenance time; and
- (vi) selectable bias towards routes that maximise production in the short-term.

The route search procedure takes these factors into account by assigning weights to each factor, depending upon their importance as specified by the user.

The SPS 1 displays candidate routes on the computer system 2, allowing the user to browse, compare and sort routes based on various criteria. The display can include the simulated movement of the ship 101 along the routes, showing projected tidal and other conditions, as described below. This assists the user to manually select a preferred route by considering the trade-offs involved. For example, the SPS 1 might determine two possible routes for a survey:

Route 1: Complete priority lines by 20/6, finish survey on 28/6

Route 2: Complete priority lines by 22/6, finish survey on 27/6

Either route may be chosen as the best route, depending on the needs of the user requiring the survey.

- 7 -

The enormous number of possible paths excludes the possibility of considering all possible paths through a survey. For example, in a survey with n lines which may be surveyed in either direction, there are $n! 2^n$ possible paths. For a twenty line survey, there are 2.5×10^{24} possible paths. Consequently, the procedure examines only a subset of all possible paths.

5

The search procedure includes two different search procedures: a short search procedure, and a tendrill search procedure. The word tendrill refers to a path consisting of survey lines joined by EOL run out 401, EOL turn 405, straight 403, SOL turn 406, and SOL run in 402 segments. The short search procedure searches from an end-of-line (EOL) point 408 for a number x of the nearest new (*i.e.*, unvisited) SOL points 410, and for each of these generates every possible path up to n steps long. The best z paths are stored in a buffer. Execution time is approximately proportional to x^n . The short search is used to find short-term paths and to build a set of seed paths for a tendrill search.

10

15 The tendrill search generates paths over any number of lines by only searching a subset of promising paths. The search is seeded with a short search, as described above, and the z resultant partial paths are extended, one survey line at a time, in parallel. At each extension step, each path is extended to a number y of nearest adjacent unvisited SOL points 410. Each extension incurs a penalty related to the time it takes to traverse the path between the EOL point 408 and the new SOL point 410, plus a time-dependent bias factor associated with the SOL point 410, as described below. Only the z paths with the lowest accumulated penalties are kept, and the extension is continued until the set of survey lines is complete. For a survey with N lines, execution time is approximately proportional to $z y N$. Up to z complete paths are produced.

20

25

The search procedures are implemented as software modules written in the object-oriented language C++. The source code for these and supporting modules is given in the Appendix below. The searches are performed on a data set derived from the survey definition data. The software modules define a number of object classes and methods, as described below.

30

- 8 -

SEARCHDATA class objects are each used to store a complete set of TurnLists and Biases, indexed by point number. Point refers to the end point of a seismic survey line. A unique point number is assigned to each end of each line.

- 5 TurnList is a list of TURNLIST objects, indexed by EOL point number, where TURNLIST class objects represent the turns available from an EOL point. Turnlists include two data members, indexed by turn rank: SOL, an array of turn indices, and Duration, a corresponding array of turn durations. The term rank represents the ranking of the turns on the basis of duration, with turns of shorter duration having a lower (*i.e.*, better) rank than
10 longer turns.

- BiasList is a list of BIAS objects, indexed by SOL point number, where BIAS class objects represent penalties applied to SOL points, and are used to bias a search towards or away from an SOL point. Penalties (which may effectively be bonuses) associated with starting
15 a line at a particular time, are calculated from feather, line or priority data. If an SOL is reached at an unfavourable time (for example, when feather on a line is predicted to exceed a predetermined limit), the penalty added to the turn time makes the path less competitive relative to other paths. The data members of the class include TimeBias, a set of integers representing time-dependent biases at 30 minute intervals, and FixedBias, an integer
20 representing a fixed bias, such as line priority or isolated lines. The method GetBias (long int Time) returns the sum of the fixed and time-dependent biases for an SOL at the specified time.

- TENDRIL class objects represent complete or partial paths through a survey, and include
25 the following data members:

- | | | |
|----|----------------|--|
| 30 | Path | an array of integers representing the indices of a sequence of points |
| | Visited | an array of booleans indicating which points have been visited by this tendril |
| | Step | an integer giving the number of points visited so far |
| | Time | a long integer giving the elapsed time from the start of the path |
| | Score | a long integer giving the paths summed turn duration and bias. |
| | pPoints | a pointer to a SEARCHDATA object |

- 9 -

TENDRIL class objects use the following methods during a search:

5 **GrowTendrilo (integer n)** Try to extend the tendril by one line, by turning to the n^{th} closest SOL. If this SOL has been visited already by this tendril, GrowTendrilo() returns 0, otherwise it returns the new value of **Score**.

PruneTendrilo Prune the tendril of its last visited line.

10 These methods automatically update Path, Visited, Step, Time and Score. Tendrils of similar **Step** may be compared on the basis of their **Score**. A lower **Score** generally indicates that the tendril has had shorter line changes, but may also indicate completed priority lines, good feather projections, etc.

15 **InsertMaintenance(long n, long t)**
Schedules a turn of a minimum duration (t minutes) within the first n lines of the tendril's path. If a turn longer than specified already exists, then no change is made; otherwise, the longest turn is extended to the specified duration.
This function is called after n lines during the Short Search.

20 **TENDRILBUFFER** objects hold lists of related tendrils of the same step number. The only data member is **Tendrilo**, which is a list of tendrils. **TENDRILBUFFER** objects use the method: **InsertTendrilo (TENDRIL NewTendrilo)**, which adds a tendril to the buffer, replacing the tendril with the worst score.

25 The **TENDRILSET** class oversees and manages the search process. A TendrilSet has the following data members:

	SearchData	a SEARCHDATA object representing turns and biases.
30	Buffer1	a TENDRILBUFFER object.
	Buffer2	a TENDRILBUFFER object.

- 10 -

	pBestBuffer	a pointer to either Buffer1 or Buffer2.
	pWorkBuffer	a pointer to either Buffer1 or Buffer2.
	NumSteps	an integer representing the number of lines to complete.
5	NumTendrils	an integer representing the number of tendrils.
	MaxShoots	the maximum number of new SOLs from any one EOL.
	MaxRank	the maximum turn rank to be attempted from any one EOL.

The **TENDRILSET** class uses the following methods:

10

ShortSearch() generates a set of seed paths of a specified but limited number of steps.

15

TendrilsSearch() generates a set of paths of any number of steps from seed paths generated by the **ShortSearch()** method.

20

GrowAllTendrils() is the main function called by **TendrilsSearch()**. It causes each Tendril in the **WorkBuffer** to extend itself to a number of adjacent points in turn, submitting the resulting extended tendrils to the **BestBuffer**.

SortBestTendrils() is a simple sort that sorts the resulting paths in order of merit.

A variety of parameters can be used to affect the behavior of the search. Some of these parameters influence the construction of the **SearchData** object, and some of them
25 influence the search itself. The most significant factor influencing the search is the number of Tendrils maintained, **NumTendrils**. The larger the number of Tendrils, the greater the diversity of the paths and the more "long sighted" the search.

MaxShoots limits the number of growths made from each tendril at each EOL point. For
30 example, if **MaxShoots** = 4, then (at most) each tendril will grow to the four closest

- 11 -

unvisited SOL points. This parameter effects the "diversity" of the search, defining the rate at which the set of tendrils 'fans out'.

MaxRank limits the Rank of turns tried at each point. For example, if MaxRank is 30, no
5 turns higher than rank 30 will be tried from any point.

A, B, C, D, E and **F** affect the Short Search used to generate seed paths.

"A" defines how many of the available choices are tried from the start position;

"B" defines how many of the available choices are tried from the end of the first visited
10 line;

"C" defines how many of the available choices are tried from the end of the second visited
line; and so on.

The product A.B.C.D.E.F determines how long the short search will take.

15 These values can be increased for complex searches involving time-dependent conditions
or maintenance, giving a wider range of potential paths examined.

In the Tendril Search, a large number of paths effectively compete with one another. In a
completely unbiased search, the attractiveness of an SOL depends solely on how long it
20 will take to get to it from the current EOL, measured in minutes. Bias factors applied to an
SOL make the choice more or less attractive relative to other points, or other times.
Negative biases make points more attractive and vice versa.

Priority bias defines the advantage given to priority lines. If a line is marked as priority,
25 this value is subtracted from the fixed bias value associated with both of its potential SOL
points. This makes the point more attractive to the Tendril Search, and causes the line to
appear earlier in the resulting paths. This value is entered directly in the search dialog.

Isolation bias is a means of making the search visit isolated lines. Although it may be
30 efficient to visit an isolated line early in a path, it may get ignored because of the
inherently shortsighted competition between tendrils. The isolation bias makes isolated

- 12 -

lines more attractive (and therefore more likely to be visited) by subtracting a proportion of the shortest turn to the point from its fixed bias. This has the effect of making all turns to the SOL equally more attractive. This percentage proportion is the **foresight parameter** referred to in the search dialog box generated by the SPS 1.

5

Time dependent biases alter the likelihood of starting a line at a particular times. They are calculated by a **Time Bias Function**, and are used to account for tidal, feather, priority or other time-dependent factors. Time dependent biases are calculated at half-hourly intervals and stored in the SOL's TimeBias set as integers prior to a search.

10

Feather Bias is calculated for a particular SOL and time by determining the number of minutes the projected feather will exceed a feather limit during the shooting of the line. The results are multiplied by the **Feather Bias Weighting** to give the feather component of the Time Bias Function.

15

Tidal Bias is calculated for lines where there is overlap between the line and tidal hazards (reefs, shallow water etc). Tidal Bias is applied if the specified SOL time would cause an active tidal hazard to be traversed. Tidal bias is multiplied by a **Tidal Bias Weighting** to give the tide component of the Time Bias Function.

20

Priority Bias is calculated for lines with a specified "before" time by giving a bonus to SOLs before and a penalty after the specified time to give the priority component.

25

The SPS 1 can be interfaced with the onboard navigation and positioning systems (e.g., a global positioning system (GPS)) of the ship 101 to obtain periodically updated information regarding ship position and orientation, feather, and survey progress. This allows the SPS 1 to generate an accurate 'live' display of the current position of the ship 101 as it traverses a route through a survey pattern, as shown in Figure 8. The ship 101 is displayed as a black triangle 802, and a status window 804 displays the current survey line identifier, the location of the ship 101, the expected feather conditions, the tide conditions, and the fraction of the total survey that has been completed. By using a pulldown menu,

30

- 13 -

the user can specify a time in the past or the future, and the SPS 1 then provides a similar display for the specified date and time. The expected location of the vessel is determined from input values for the speed of the vessel on survey lines, during line change turns, and during line change straight segments. Knowledge of the ship's current position and the selected survey route also allows the SPS 1 to generate navigational data for navigation of the ship 101.

The SPS 1 includes a number of further display options for the user. As shown in Figure 9, the predicted feather conditions along each survey line can be displayed as coloured wedges 902 along each survey line. A tide properties window 904 can be displayed, providing a graphic image showing tide conditions, including a numeric value for the current tide conditions, and a graphic representation of the variation in tide conditions for the previous and future twelve hours. Daytime and nighttime are indicated by white 906 and black 908 regions above a cross hatched area 910 representing the ocean.

Submerged hazards are an important consideration for planning survey routes. As shown in Figure 10, the SPS 1 displays the expected clearance over submerged hazards 1002, 1004. A safe hazard over which there is sufficient clearance to safely pass is displayed as a yellow circle 1002, whereas an unsafe hazard over which is not safe to pass is displayed as an orange circle 1004. The tide properties display 904 provides the user with an indication of the relative tide conditions at the corresponding time and date.

The SPS 1 allows the user to determine completion times for potential new survey designs, depending on the vessel and equipment configuration used. Thus the user can compare alternative vessel and equipment configurations at the survey design or tendering stage in terms of survey duration and cost.

Many modifications will be apparent to those skilled in the art without departing from the scope of the present invention as herein described with reference to the accompanying drawings.

- 14 -

APPENDIX

```

int    CTENDRILSET::ShortSearch()
5      {
        //Generates a set of paths 6 lines long
        //can be more by adding extra loops

        //pBestBuffer is a pointer to a TENDRILBUFFER where the search
        //results will be stored
10      //reset the best buffer
        pBestBuffer->ClearBuffer();
        //pBestBuffer contains NumTendrils blank tendrils

15      //integers a,b,c,d,e,f in loops represent turn rank numbers
        //for example, a=2,b=1,c=4,d=5,e=0,f=3
        //will create a tendril that attempts to
        //take the 3rd 'closest' SOL Point from the current location
        //then the 2nd 'closest' SOL Point to the location at end of step 1
20      //then the 5th 'closest' SOL Point to the location at end of step 2
        //then the 6th 'closest' SOL Point to the location at end of step 3
        //then the 1st 'closest' SOL Point to the location at end of step 4
        //then the 4th 'closest' SOL Point to the location at end of step 5

25      //the following set of loops builds a tendril for all possible
        //paths within the parameters.
        //if an SOL Point has already VISITED by this tendril, the
        //tendril is aborted when Tendril.GrowTendrilTo(n) returns 0.

30      A=8;B=6;C=6;D=6;E=6;F=4;

        for (int a=0;a<A;a++) //at any FIRST TURN 8 choices attempted
        for (int b=0;b<B;b++)
        for (int c=0;c<C;c++) //these are
35      for (int d=0;d<D;d++)
        for (int e=0;e<E;e++)
        for (int f=0;f<F;f++) //at any SIXTH TURN 4 choices attempted

        {
40          //create a new, blank Tendril as a local variable
          CTENDRIL Tendril;
          //reset it
          Tendril.ClearTendril();

          //grow Tendril to the ath closest SOL
          if (Tendril.GrowTendrilTo(a))

45              //if 1st step's choice a was valid
              if (Tendril.GrowTendrilTo(b))

50                  //if 2nd step's choice b was valid
                  if (Tendril.GrowTendrilTo(c))

55                      //if 3rd step's choice c was valid
                      if (Tendril.GrowTendrilTo(d))

                          //if 4th step's choice d was valid
                          if (Tendril.GrowTendrilTo(e))

60                              //if 5th step's choice e was valid
                              if (Tendril.GrowTendrilTo(f))

                                  //if the tendril made it to step 6
                                  //attempt to add the Tendril to best buffer
65                                  pBestBuffer->Insert(Tendril);
        }

        //pBestBuffer now contains NumTendrils good starting paths 6 lines long

```

- 15 -

5),
 return(12) //12 is the current step number - 12 points (6 lines) visited
 //nb : calls to InsertMaintenance removed for clarity
 //eg
 //if Nth step's choice X was valid
 // if (MaintainBefore == N)
 // InsertMaintenance(N, MaintenanceTime)
 //

- 16 -

```
void CTENDRILSET::TendrilsSearch()
{
5   //build the point set and turn ranking database
   //using info in the survey definition
   SearchPoints.CreateTurnSet(&SURVEYDEFINITION);

10  //calculate time dependent bias look up table for all points
   SearchPoints.CreateBiasSet(&SURVEYDEFINITION);

   //perform a Short Search to build seed tendrils
   StepNumber = ShortSearch();

15  //keep growing Tendrils one step at a time until
   //desired number of steps performed
   while (StepNumber < NumSteps)

       StepNumber = GrowAllTendrils();

20  //sort the tendrils into order of rank
   SortBestTendrils ();

   //pBestBuffer now contains a set of tendrils of length NumSteps
   //that represent quasi optimal paths

25 }
```

- 17 -

```

int CTENDRILSET::GrowAllTendrils()
{
5   //last steps results are at pBestBuffer, so swap the two pointers
   swap (pBestBuffer,pWorkBuffer);

   //pBestBuffer : empty
   //pWorkBuffer : contains the best tendrils from the previous step
10  //grow all the TENDRILS to a restricted number of adjacent points
   //extended TENDRILS are submitted for inclusion in pBestBuffer

   for (int Counter=0;Counter<NumTendrils;Counter++)
15   {
       //NewRank is the Rank number of the turn being attempted
       int NewRank = 0;

       //New Shoots counts number of successful attempts
20       int NewShoots = 0;

       do
       {
           //note GrowTendrillTo method returns 0 if turn was invalid
           //for example when the attempted turns SOL POINT has already
25           //been visited by this tendril

           //attempt to make turn of rank [choice]
           if (pWorkBuffer->Tendrill[Counter].GrowTendrillTo(NewRank)>0)

30           //if the shoot was grown successfully,
           {
               //offer the now extended tendril to Best
               pBestBuffer->Insert(pWorkBuffer->Tendrill[Counter]);

35               //undo the turn ready for the next attempt
               pWorkBuffer->Tendrill[Counter].PruneTendrill();

               //a shoot was grown, so
               NewShoots++;
40           }

           //all growths successful or not use a choice so
           NewRank++;
45       }

       while ((NewShoots < MaxShoots) && (NewRank < MaxRank));

       //reset the working tendril so that it can be reused
       pWorkBuffer->Tendrill[Counter].ClearTendrill ();
50   }

   //pBestBuffer : NUMTENDRILS unsorted best tendrils with STEP N+1
   //pWorkBuffer : Empty
55   return(pBestBuffer->Tendrill[0].Step)
}

```

- 18 -

```

longint CTENDRIL::GrowTendrillTo(int nextchoice)
{
5    //causes a tendrill to extend itself by taking
    //the turn of rank nextchoice.
    //eg next choice = 2 will attempt the 3rd shortest line change
    //if the point to be turned to has been visited, returns 0
    //otherwise returns the new accumulated tendrill score

10    //Get a pointer to this Tendrills current EOL point's turn list
    TURNLIST *TurnList = &pPoints->TurnList[CurrentPoint()];

    //NewSOLPoint is the index of the next SOL POINT
15    int NewSOLPoint = TurnList->SOL[nextchoice];

    //if this new SOL point has already been visited, abort this tendrill
    if (Visited(NewSOLPoint)) return(0);

20    //Find this new turns duration from the current EOL point's Turn List
    int TurnDuration = TurnList->Duration[nextchoice];

    //Add the turn duration to the tendrills current time to get the new SOL time
    Time += TurnDuration;

25    //GetBias(CurrentTime) returns the sum of the fixed bias
    //and the time dependent bias associated with this SOL time -
    //if now is a bad time to travel to the newSOLPoint, eg due to high feather
    //GetBias will return a high value, penalizing this tendrill

30    //find bias associated with the new SOL point and time
    int Bias = pPoints->BiasList[NewSOLPoint].GetBias(CurrentTime);

    int NewStep = Step + 1;

35    //Path is an array of EOL / SOL indexes, represents the PATH of the tendrill
    Path[NewStep] = NewSOLPoint;

    //mark the SOLPoint as visited by this tendrill
    SetVisited(NewSOLPoint,true);

40    //add the cost of the turn
    Score = Score + TurnDuration + FixedBias + TimeBias;

    //add the duration of the new Line
45    Time += LineDuration[NewSOLPoint];

    //the new EOL is the SOL index XOR 1, the opposite end of the Line
    int NewEOLPoint = NewSOLPoint ^ 1;

50    NewStep++;
    Path[NewStep] = NewEOLPoint;
    Step = NewStep;
    SetVisited(NewEOLPoint,true);

55    return(Score);

    //note : the storage of tendrills is inefficient (representing both ends of a line
    //in the path is actually redundant and may be refined)
}

```

- 19 -

CLAIMS:

1. A method for use in generating at least one route traversing a plurality of predetermined seismic survey paths in a marine seismic survey, including:
 - 5 generating a plurality of routes traversing at least a subset of said predetermined seismic survey paths, said routes satisfying predetermined constraints;
 - generating respective penalties for said routes; and
 - selecting at least one of said routes on the basis of said penalties.
- 10 2. A method for use in generating at least one route traversing a plurality of predetermined seismic survey paths in a marine seismic survey, including:
 - (a) generating a plurality of partial routes, each partial route including a different subset of said predetermined survey paths, said partial routes satisfying predetermined constraints;
 - 15 (b) for each partial route, generating a set of corresponding extended routes by extending the partial route to include respective selected ones of remaining predetermined seismic survey paths, said extended routes satisfying predetermined constraints;
 - (c) generating penalties for said extended routes;
 - 20 (d) selecting a subset of said extended routes on the basis of said penalties; and
 - (e) repeating steps (b)–(d), using said subset of extended routes as said set of partial routes.
- 25 3. A method as claimed in claim 2, wherein each partial route includes the same number of said survey paths.
4. A method as claimed in claim 2, wherein said step of repeating steps (b)–(d) includes repeating said steps until each of said extended routes includes all members of said plurality of predetermined seismic survey paths.
- 30 5. A method as claimed in claim 2, wherein said step of repeating steps (b)–(d) includes repeating said steps until said set of extended routes includes all but a predetermined

- 20 -

number of said plurality of predetermined seismic survey paths, and including the steps of:

for each extended route, further extending said route to include possible partial routes traversing remaining members of said plurality of predetermined seismic survey paths, said partial routes satisfying predetermined constraints;

generating penalties associated with each of said extended routes; and

selecting a subset of said extended routes on the basis of the penalty associated with each of said extended routes.

6. A method as claimed in claim 2, wherein said selected ones of said plurality of predetermined seismic survey paths are selected on the basis of distances between the end of the partial route and ends of said predetermined seismic survey paths.

7. A method as claimed in claim 2, wherein said step of generating a set of partial routes includes:

selecting, from a predetermined point, a plurality of start points, each start point corresponding to a respective one of said seismic survey paths;

generating, from each of said start points, possible partial routes traversing a predetermined number of said seismic survey paths, said partial routes satisfying predetermined constraints;

generating penalties for said partial routes; and

selecting a subset of said partial routes on the basis of said penalties.

8. A method as claimed in claim 1, including:

generating routes traversing said plurality of predetermined seismic survey paths from the selected at least one of said routes;

displaying said routes for comparison; and

selecting at least one route on the basis of said comparison.

9. A method as claimed in claim 1, wherein said constraints include run in, run out and turn constraints.

- 21 -

10. A method as claimed in claim 1, wherein said penalties include penalties for path components of said partial routes.
11. A method as claimed in claim 1, wherein said penalties include penalties for line changes between survey paths.
12. A method as claimed in claim 1, wherein said step of generating penalties includes generating penalties on the basis of at least one of time, distance and location.
13. A method as claimed in claim 1, wherein said step of generating penalties includes generating penalties on the basis of respective predetermined priorities for selected paths of said predetermined survey paths.
14. A method as claimed in claim 1, wherein said step of generating penalties includes generating one or more penalties for respective hazards.
15. A method as claimed in claim 1, wherein said step of generating penalties includes generating one or more time-dependent penalties.
16. A method as claimed in claim 15, wherein said time-dependent penalties include feather offset penalties.
17. A method as claimed in claim 15, wherein said time-dependent penalties include feather projection penalties.
18. A method as claimed in claim 15, wherein said time-dependent penalties include penalties for at least one tidal hazard
19. A method as claimed in claim 1, wherein said step of generating penalties includes generating one or more penalties based on survey completion location.
20. A method as claimed in claim 1, wherein said step of generating penalties includes generating one or more penalties based on maintenance requirements.

- 22 -

21. A method as claimed in claim 1, including determining locations for a survey vessel at future times on a selected route.

5 22. A method as claimed in claim 21, including displaying simulated movement of said vessel on said route.

23. A method as claimed in claim 21, including determining tide levels for said vessel at said times.

10

24. A method as claimed in claim 23, including determining clearances between said vessel and at least one hazard at said times.

25. A method as claimed in claim 21, including determining water currents at said times.

15

26. A method as claimed in claim 25, including determining streamer feather values at said times.

27. A method as claimed in claim 23, including displaying tidal data.

20

28. A method for evaluating a seismic survey comprising a plurality of predetermined seismic survey paths in a marine seismic survey, including:

generating a plurality of routes traversing a subset of said predetermined seismic survey paths, said routes satisfying predetermined constraints;

25

generating respective penalties for said routes;

selecting at least one of said routes on the basis of said penalties;

generating a route traversing said plurality of predetermined seismic survey paths from the selected at least one of said routes; and

evaluating said survey on the basis of a metric of said route.

30

29. A method as claimed in claim 28, wherein said metric includes at least one of survey duration and survey cost.

- 23 -

30. A method as claimed in claim 28, including evaluating a number of surveys and determining metrics for said surveys, respectively, and selecting one of said surveys on the basis of said metrics.

5 31. A method as claimed in claim 30, wherein said surveys have differences in at least one of vessel configuration, streamer configuration, and survey pattern.

32. Marine seismic survey software having program code for executing the steps of any one of claims 1 to 31.

10

33. A computer readable storage medium having stored thereon program code for executing the steps of any one of claims 1 to 31.

34. A marine seismic survey system having components for executing the steps of any one of claims 1 to 31.

15

35. A marine seismic survey system as claimed in claim 32, including an interface for receiving location data for a survey vessel.

20 36. A system for controlling a marine seismic survey, including:

an interface to a positioning system for determining the current position of a survey vessel; and

a marine seismic survey system as claimed in claim 33 including means for generating navigational data for said survey vessel based on said current position to

25

traverse said route.

37. A system as claimed in claim 36, wherein said positioning system is a GPS.

38. A method for use in generating at least one route traversing a plurality of predetermined survey paths in a marine survey, including:

generating a plurality of routes traversing at least a subset of said predetermined survey paths, said routes satisfying predetermined constraints;

5 generating respective penalties for said routes; and

selecting at least one of said routes on the basis of said penalties.

39. A method for use in generating at least one route traversing a plurality of predetermined survey paths in a marine survey, including:

10 (a) generating a plurality of partial routes, each partial route including a different subset of said predetermined survey paths, said partial routes satisfying predetermined constraints;

(b) for each partial route, generating a set of corresponding extended routes by extending the partial route to include respective selected ones of remaining
15 predetermined survey paths, said extended routes satisfying predetermined constraints;

(c) generating penalties for said extended routes;

(d) selecting a subset of said extended routes on the basis of said penalties; and

(e) repeating steps (b)-(d), using said subset of extended routes as said set of
20 partial routes.

40. A method as claimed in claim 39, wherein each partial route includes the same number of said survey paths.

25 41. A method as claimed in claim 39, wherein said step of repeating steps (b)-(d) includes repeating said steps until each of said extended routes includes all members of said plurality of predetermined survey paths.

42. A method as claimed in claim 39, wherein said step of repeating steps
30 (b)-(d) includes repeating said steps until said set of extended routes includes all but a predetermined number of said plurality of predetermined survey paths, and including the steps of:

for each extended route, further extending said route to include possible partial routes traversing remaining members of said plurality of predetermined survey paths, said partial routes satisfying predetermined constraints;

generating penalties associated with each of said extended routes; and

5 selecting a subset of said extended routes on the basis of the penalty associated with each of said extended routes.

43. A method as claimed in claim 39, wherein said selected ones of said plurality of predetermined survey paths are selected on the basis of distances between
10 the end of the partial route and ends of said predetermined survey paths.

44. A method as claimed in claim 39, wherein said step of generating a set of partial routes includes:

selecting, from a predetermined point, a plurality of start points, each start
15 point corresponding to a respective one of said survey paths;

generating, from each of said start points, possible partial routes traversing a predetermined number of said survey paths, said partial routes satisfying predetermined constraints;

generating penalties for said partial routes; and

20 selecting a subset of said partial routes on the basis of said penalties.

45. A method as claimed in claim 38, including:

generating routes traversing said plurality of predetermined survey paths from the selected at least one of said routes;

25 displaying said routes for comparison; and

selecting at least one route on the basis of said comparison.

46. A method as claimed in claim 38, wherein said constraints include run in, run out and turn constraints.

47. A method as claimed in claim 38, wherein said penalties include penalties for path components of said partial routes.

48. A method as claimed in claim 38, wherein said penalties include
5 penalties for line changes between survey paths.

49. A method as claimed in claim 38, wherein said step of generating penalties includes generating penalties on the basis of at least one of time, distance and location.
10

50. A method as claimed in claim 38, wherein said step of generating penalties includes generating penalties on the basis of respective predetermined priorities for selected paths of said predetermined survey paths.

51. A method as claimed in claim 38, wherein said step of generating
15 penalties includes generating one or more penalties for respective hazards.

52. A method as claimed in claim 38, wherein said step of generating penalties includes generating one or more time-dependent penalties.
20

53. A method as claimed in claim 52, wherein said time-dependent penalties include feather offset penalties.

54. A method as claimed in claim 52, wherein said time-dependent
25 penalties include feather projection penalties.

55. A method as claimed in claim 52, wherein said time-dependent penalties include penalties for at least one tidal hazard.

56. A method as claimed in claim 38, wherein said step of generating
30 penalties includes generating one or more penalties based on survey completion location.

57. A method as claimed in claim 38, wherein said step of generating penalties includes generating one or more penalties based on maintenance requirements.

5 58. A method as claimed in claim 38, including determining locations for a survey vessel at future times on a selected route.

59. A method as claimed in claim 58, including displaying simulated movement of said vessel on said route.

10

60. A method as claimed in claim 58, including determining tide levels for said vessel at said times.

15 61. A method as claimed in claim 60, including determining clearances between said vessel and at least one hazard at said times.

62. A method as claimed in claim 58, including determining water currents at said times.

20 63. A method as claimed in claim 62, including determining streamer feather values at said times.

64. A method as claimed in claim 60, including displaying tidal data.

25 65. A method for evaluating a survey comprising a plurality of predetermined survey paths in a marine survey, including:

generating a plurality of routes traversing a subset of said predetermined survey paths, said routes satisfying predetermined constraints;

generating respective penalties for said routes;

30 selecting at least one of said routes on the basis of said penalties;

generating a route traversing said plurality of predetermined survey paths from the selected at least one of said routes; and

evaluating said survey on the basis of a metric of said route.

66. A method as claimed in claim 65, wherein said metric includes at least one of survey duration and survey cost.

5

67. A method as claimed in claim 65, including evaluating a number of surveys and determining metrics for said surveys, respectively, and selecting one of said surveys on the basis of said metrics.

10 68. A method as claimed in claim 67, wherein said surveys have differences in at least one of vessel configuration, streamer configuration, and survey pattern.

15 69. Marine survey software having program code for executing the steps of any one of claims 38 to 68.

70. A computer readable storage medium having stored thereon program code for executing the steps of any one of claims 38 to 68.

20 71. A marine survey system having components for executing the steps of any one of claims 38 to 68.

72. A marine survey system as claimed in claim 69, including an interface for receiving location data for a survey vessel.

25

73. A system for controlling a marine survey, including:
an interface to a positioning system for determining the current position of a survey vessel; and

30 a marine survey system as claimed in claim 70 including means for generating navigational data for said survey vessel based on said current position to traverse said route.

- 29 -

74. A system as claimed in claim 73, wherein said positioning system is a GPS.