Research Article

Dynamic Games Methods in Synthesis of Safe Ship Control Algorithms

Józef Lisowski

Department of Ship Automation, Faculty of Electrical Engineering, Gdynia Maritime University, 81-225 Gdynia, Poland

Correspondence should be addressed to Józef Lisowski; j.lisowski@we.am.gdynia.pl

Received 16 May 2018; Accepted 16 September 2018; Published 1 October 2018

Academic Editor: Alain Lambert

Copyright © 2018 Józef Lisowski. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The paper presents application of dynamic games methods, multistage positional and multistep matrix games, to automate the process control of moving objects, on the example of safe control of own ship in collision situations when passing many ships encountered. Taking into consideration two types of ships cooperation, for each of the two types of games, positional and matrix, four control algorithms for determining a safe ship trajectory supporting the navigator’s maneuvering decision in a collision situation are presented. The considerations are illustrated by examples of computer simulation in Matlab/Simulink software of safe trajectories of a ship in a real situation at sea. Taking into account the smallest final deviation game trajectory from the reference trajectory of movement, in good visibility at sea, the best is trajectory for cooperative matrix game, but in restricted visibility at sea, the best is trajectory for cooperative positional game.

1. Introduction

One of the most important transport issues are the processes of optimal and safe control of ships, airplanes, and cars as moving objects [1–4]. Such processes relate to managing the movement of many objects at the same time, with varying degrees of interaction, the impact of random factors with an unknown probability distribution, and a large share of operator’s subjective in maneuvering decisions [5–8]. Therefore, the management of such processes is accomplished by means of game control systems, whose synthesis is carried out with the methods of game theory [9, 10]. Game theory is a branch of mathematics, covering the theory of conflict situations and building and analyzing their models [11, 12]. Conflict can be as follows: military, political, social, and economic, in a social game, in the game with nature, and in the implementation of the control process during interferences of disturbances or other control objects [13]. A game in the concept of control theory is a process consisting of several control objects remaining in a conflict situation or a process with undefined disturbances or incomplete information. Players as control objects participating in a conflict situation have certain sets of strategies. Strategy is a set of rules of action, player control, which cannot change the actions of an opponent or nature [14–16]. The strategies are implemented by man, automaton, regulator, and computer. Strategies can be pure as elements of a set of strategies or mixed as a probability distribution on a set of clean strategies. The result of the game is the payment in the form of winning, losing, or the probability of carrying out a certain action—control [12, 17–19].

The first concept of game theory and the theorem on mini-max was formulated by E. Borel (1921, 1927). The main creators of game theory are John von Neumann (1928) and O. Morgenstern (1944).

The largest class of games that can be used in the game control of dynamic transport processes and among them controlling the movement of ships, planes, and cars represent differential games, described by state and output equations, and state and control constraints [20–24].

2. Classification of Control Processes of Moving Objects

As a result of the movement of own ship with speed $V$ and course $\psi$ in terms of encountered $j$ ship moving at a speed $V_j$ and course $\psi_j$ a situation at sea is determined. Parameters characterizing the situation as distance $D_j$ and bearing $N_j$ for $j$ ship are measured by radar anticollision system ARPA (Automatic Radar Plotting Aids) [25].

The ARPA system enables us to track automatically at least 20 encountered $J$ ships, determine of their movement parameters ($V_j$, $\psi_j$) and elements of approach to the own ship ($D_{j \text{min}}$, $T_{j \text{min}}$), and also assess the collision risk (see Figure 1).

The proper use of anticollision system ARPA in order to achieve greater safety of navigation requires, in addition to training on the use and interpretation of the data, supplement the system with appropriate methods of computer-aided maneuvering decision of navigator in the complex navigational situation in a short time, eliminating the subjectivity of man and taking into account the indefiniteness of the situation and the properties game process control [26, 27].

In practice, there are many possible maneuvers to avoid a collision, from which to select the optimal maneuver, to ensure a minimum the risk of collision or minimum losses of the road for safe passage of encountered ships (see Figure 2).

The movement of objects in time is influenced by control variables $u$ from appropriate admissible control sets $U$:

$$ u \in U \left( U_0^{(0)}, U_j^{(0)} \right), $$

where

- $U_0^{(0)}$ set of strategies for own object,
- $U_j^{(0)}$ set of strategy $j$ of encountered object from among the total number of $J$ objects,
- $\theta = 0$ symbolically means stabilization of the set object trajectory,
- $\theta = 1$ symbolically the implementation of an anticollision maneuver to minimize the risk of collision, which in practice is achieved by meeting inequalities:

$$ D_{j \text{min}} = \min D_j(t) \geq D_s, \quad j = 1, 2, \ldots, J $$

$D_{j \text{min}}$: smallest distance of approaching own object to the object you are meeting,
$D_s$: safe proximity distance in given ambient conditions, traffic rules, and dynamic properties of the object,

- $\theta = -1$: symbolic maneuvering the object in order to achieve the shortest approaching distance, for example, when transferring the load.

Following types of motion control objects can be distinguished:

(1) Conflict Games:

(i) situations of unilateral dynamic game: $U(U_0^{(0)} U_j^{(0)})$ and $U(U_0^{(0)} U_j^{(0)});$

(ii) chase situations: $U(U_0^{(-1)} U_j^{(-1)})$ and $U(U_0^{(-1)} U_j^{(-1)});$

(2) Unilateral Games:

(i) avoiding collisions with

(1) maneuvers of own ship: $U(U_0^{(1)} U_j^{(0)});$  
(2) maneuvers of $j$ object encountered: $U(U_0^{(0)} U_j^{(1)});$  
(3) cooperating maneuvers: $U(U_0^{(1)} U_j^{(1)});$  

(ii) meeting of objects: $U(U_0^{(-1)} U_j^{(-1)});$  

(3) Optimal Control:

(i) stabilization of reference trajectory of object movement: $U(U_0^{(0)} U_j^{(0)}).$
where the form (4).

relative motion of own ship and ship encountered, will take own ship’s hydromechanics equations and the kinematics of control processes in collision situations, taking into account the form (4).

For example, the equations of the state of the own ship by the state equation:

\[ J = 2 \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]

\[ = a x_0 + b u, \]
3.2. Positional Game Model. The differential game model comes down to a multistage positional game model, in which the object’s dynamics are taken into account by the time of the manoeuvre ahead of time. The essence of positional game is dependence of own object’s strategy on the positions of the objects \( p(t) \) taught. In this way, possible changes in the course and speed of the objects encountered during the control implementation are taken into account in the process model. The current state of the process at the moment \( t_k \) is determined by the coordinates of own ship position \( x_0 \) and encountered objects \( x_j \):

\[
\begin{bmatrix}
    x_0(t_k) \\
    x_j(t_k)
\end{bmatrix} 
= p(t_k), \quad (j = 1, 2, \ldots, J), \quad (k = 1, 2, \ldots, K).
\]

(7)

It is assumed that at each discrete moment of time the own ship position and the met objects positions are known. Constraints of state variables are navigational constraints on the surrounding of encountered objects:

\[
\begin{bmatrix}
    x_0(t) \\
    x_j(t)
\end{bmatrix} \in P,
\]

(8)

Constraints of control variables take into account the motion kinematics of objects, legal recommendations for traffic regulations (maritime traffic law, air traffic law, and road code), and the condition of maintaining a safe passing distance:

\[
u_0 \in U_0, \\
u_j \in U_j \quad (j = 1, 2, \ldots, J).
\]

(9)

The sets of acceptable strategies of the players in relation to each other are dependent, which means that the choice of control by the \( j \)th of the object changes the sets of acceptable strategies of other objects:

\[
\{U^j_i [p(t)], U^0_j [p(t)]\},
\]

(10)

The resultant area of acceptable manoeuvres of own object in relation to \( J \) objects is

\[
U_0 = \bigcap_{j=1}^J U^0_j \quad (j = 1, 2, \ldots, J),
\]

(11)

Optimal control of the own object, ensuring minimal road loss on safe passing of the objects encountered, is determined by the static optimization method from the set of permissible controls \( U_0^* \):

\[
u_0^* \in U_0.
\]

(12)

3.3. Matrix Game Model. The differential game model comes down to a multistep matrix game model, in which the object’s dynamics are accounted for by the time of manoeuvre ahead. The game matrix \([r_j(s_0, s_j)]\) contains collision risk values \( r_j \) determined for permissible strategies \( s_0 \) of own object and acceptable strategies \( s_j \) of encountered \( j \)th object. Collision risk value is defined as a reference to the current approximation situation, described by objects close-up parameters \( D_{min}^j \) and \( T_{min}^j \) to the assumed assessment of the situation as safe, determined by the safe distance proximity \( D_s \) and safe time \( T_s \), necessary for the collision avoidance manoeuvre and distance \( D_j \):

\[
r_j = \left[ \zeta_1 \left( \frac{D_{min}^j}{D_s} \right) + \zeta_2 \left( \frac{T_{min}^j}{T_s} \right) + \zeta_3 \left( \frac{D_j}{D_s} \right) \right]^{-0.5}, \quad (13)
\]

where

\[
\zeta_1, \zeta_2, \text{ and } \zeta_3 \text{ are coefficients depending on the state of object movement environment.}
\]

In a matrix game, own object as player A has the ability to use \( s_0 \) different pure strategies, and \( J \) objects representing player B have \( s_j \) different pure strategies:

\[
R = r_j(s_0, s_j) = \begin{bmatrix}
    r_{1,1} & r_{1,2} & \cdots & r_{1,s_j} & \cdots & r_{1,1} \\
    r_{2,1} & r_{2,2} & \cdots & r_{2,s_j} & \cdots & r_{2,1} \\
    \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\
    r_{s_0-1,1} & r_{s_0-1,2} & \cdots & r_{s_0-1,s_j} & \cdots & r_{s_0-1,1} \\
    r_{0,1} & r_{s_0,2} & \cdots & r_{s_0,s_j} & \cdots & r_{s_0,1}
\end{bmatrix}, \quad (14)
\]

Constraints on the choice of strategy \((s_0, s_j)\) result from legal recommendations of traffic COLREGs regulations. Because usually the game has no saddle point, so there is no guaranteed balance [37, 38].

4. Game Ship Control Algorithms

The synthesis of algorithms for the control of moving objects was carried out on the example of the safe motion control process of one’s own ship during the meeting of other ships. Individual models of the process can be assigned the appropriate algorithms of computer-aided navigating maneuvering decisions in collision situations [39, 40].

The exact but complex model of differential game serves as a simulation model to check the correctness of control algorithms based on approximate positional and matrix game models.

4.1. Algorithm of Positional Noncooperative Game. The optimal control of your own ship is calculated by determining the sets of acceptable strategies for the ships you meet with respect to own ship and the sets of acceptable own ship strategy for each of the ships you meet. Then the optimal
positional strategy of the own ship is determined from the condition:

$$I^* = \min_{u_0} \max_{u_j} \min_{u_i} [x_0, P_k] = S_0^*,$$  

(15)

The goal control function of own ship $S_0$ characterizes the distance of own ship to the nearest point of return $P_k$ on a given voyage route. The criterion for choosing the optimal trajectory of own ship is to determine its course and speeds ensuring the smallest loss of the path for safe passing of encountered ships, at a distance not lower than the assumed value of $D_s$, taking into account the dynamics of own ship in the form of advance time of manoeuvre. First, the control of own ship is determined to ensure the shortest trajectory of the flight, the smallest loss of the road (min condition) for noncooperating control of every ship encountered, contributing to the largest extension of the trajectory of the own ship (max condition). At the end, from the set of controls of own ship to particular $j$ placed ships, the control of own ship is selected in relation to all $J$ ships encountered, ensuring the smallest loss of the road (condition min). According to the optimization three conditions (min max min), the linear programming method is used to solve the game, obtaining the optimal values of the course and the speed of own ship. The smallest road losses are achieved for the maximum projection of the ship's own speed vector on the course direction. Optimal control is calculated many times at each discrete stage of motion using the SIMPLEX method to solve the linear programming problem for variables in the form of components of the ship's own speed vector [41].

4.2. Algorithm of Positional Cooperative Game. For a cooperative game, the control criterion (15) will take the following form:

$$I^* = \min_{u_0} \min_{u_j} \min_{u_i} [x_0, P_k] = S_0^*.$$  

(16)

The difference in relation to the previous algorithm results from the cooperation in avoiding collision by all objects encountered $J$ and replacing the second condition max for min [42].

4.3. Algorithm of Matrix Noncooperative Game. The dual linear programming method can be used to determine the optimal control. In the dual issue, player A seeks to minimize the risk of collision, while player B in the noncooperative game aims to maximize the risk of collision [43, 44]. The components of the mixed strategy express the probability distribution of players using their pure strategies. As a result, for the control criterion in the form

$$I^* = \min_{u_0} \max_{u_j} r_j,$$  

(17)

a matrix of probabilities of using individual pure strategies is obtained.

The most secure probability of $p_j$ is the solution to the task of safe control of own ship:

$$u_0^* = u_0^0 \{ [p_j (s_0, s_j)]_{\max} \},$$  

(18)

Applying dual linear programming to matrix game solution, the optimal values of own ship course and $j$th met ship are obtained, with the smallest deviations from their initial values.

4.4. Algorithm of Matrix Cooperative Game. For a cooperative game, the control criterion (17) will take the following form:

$$I^* = \min_{u_0} \min_{u_j} r_j.$$  

(19)

The difference in relation to the previous algorithm results from the cooperation in avoiding collision by all objects encountered $J$ and replacing the second condition max for min.

5. Computer Simulation of Game Ship Control Algorithms

Figures 4, 5, 6, and 7 show the own ship safe trajectories determined by four algorithms previously in the MATLAB/SIMULINK software, in the situation of $J = 34$ encountered ships in the Kattegat Strait, in conditions of (a) good visibility at sea, (b) restricted visibility at sea for $D_s = 0.3$ nm (nautical miles) and (b) restricted visibility at sea for $D_s = 1.5$ nm.

The game ends at the moment $t_k$, when the risk of own ship $r_j$ in relation to each $j$ ship will reach the value of zero $r_j (t_k) = 0$ and then the final deviation of the trajectory of own ship from reference trajectory $d(t_k)$ is assessed.

Figure 8 compares the trajectories calculated by individual four algorithms.

In Figure 8(a), showing the safe trajectories of own ship in conditions of good visibility at sea, the best is trajectory 4 for a cooperative matrix game, providing the smallest final deviation from the reference trajectory of movement, $d(t_k) = 0.57$ nm.

In Figure 8(b), showing the safe trajectories of own ship in conditions of restricted visibility at sea, the best is trajectory 2 for a cooperative positional game, providing the smallest final deviation from the reference trajectory of movement, $d(t_k) = 1.56$ nm.

6. Conclusions

The use of simplified differential game models of the control process of moving objects, in the form of a multistage positional game and multistep matrix game, for the synthesis of control algorithms allows us to determine the safe optimal and game trajectory of own object in passing situations with more objects as a sequence of manoeuvres at a course and speed. The developed control algorithms take into account the legal rules of object movement and manoeuvre advance time, approximating the dynamic properties of the own object and assessing the final deviation of the actual trajectory from the reference one. The presented control algorithms constitute formal models of the actual decision-making processes of the ship’s navigator and can be used in the computer navigator support system when making manoeuvre decisions in collision situation.
Figure 4: Safe trajectory of own ship in the situation of passing with \( J = 34 \) met ships, determined by the positional noncooperative game algorithm.

Figure 5: Safe trajectory of own ship in the situation of passing with \( J = 34 \) met ships, determined by the positional cooperative game algorithm.

Figure 6: Safe trajectory of own ship in the situation of passing with \( J = 34 \) met ships, determined by the matrix noncooperative game algorithm.
Figure 7: Safe trajectory of own ship in the situation of passing with \( J = 34 \) met ships, determined by the matrix cooperative game algorithm.

Figure 8: Comparison of safe trajectories of own ship in the situation of passing by with \( J = 34 \) met ships, determined by individual four algorithms: (1) positional noncooperative game, (2) positional cooperative game, (3) matrix noncooperative game, and (4) matrix cooperative game; (a) conditions of good visibility at sea for \( D_s = 0.3 \) nm and (b) conditions of restricted visibility at sea for \( D_s = 1.5 \) nm.

Data Availability

The description of navigational situations used for computer calculations and their resulting data used to support the findings of this study are included within the article.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The research was supported by the Ministry of Science and Higher Education, within the framework of funds for statutory activities (Grant no. 446//DS/2018: Design and Computer Simulation Tests of Marine Automation Systems in the Matlab/Simulink and LabVIEW software) of the Electrical Engineering Faculty of Gdynia Maritime University in Poland.

References


