TRANSCOMP – XIV INTERNATIONAL CONFERENCE
COMPUTER SYSTEMS AIDED SCIENCE, INDUSTRY AND TRANSPORT

*projektowanie inżynierskie,
pojazd niewidomego pacjenta,
czujniki zbliżeniowe*

Mikołaj BASZUN[1]
Bogdan D. CZEJDO[2]

ITERATIVE DESIGN OF PROXIMITY SENSOR ARRAY FOR BLIND PATIENT VEHICLE

The paper discusses a design process for proximity sensors that can be used to steering or guide the Blind Patient Vehicle (BPV,) to obtain ultrasonic monitoring of the nearby object leading to their recognition and localization. The objective of this paper was to develop an iterative process of engineering design that can be used to produce inexpensive proximity measuring devices tailored to the specific applications. The iterative process of engineering design is based on state transition diagrams. Since one-level state transition diagrams can be quite complicated, we discuss how to simplify them by using many levels of abstraction. The resulting multi-level state diagrams are shown. We discuss how to use such multi-level diagrams to model iterative engineering design on the example of a proximity sensor. In the summary we characterize the obtained design results.

PROJEKTOWANIE ITERACYJNE TABLICOWEGO CZUJNIKA ZBLIŻENIOWEGO DLA POJAZDU NIEWIDOMEGO PACJENTA

W pracy przedstawiono proces projektowania dla tablicowego systemu czujnika zbliżeniowego, który może być użyty do sterowania lub prowadzenia pojazdu niedowidzącego pacjenta (PNP), aby uzyskać ultradźwiękowe monitorowanie pobliskich obiektów prowadzącego do ich rozpoznawania oraz lokalizowania. Celem tej pracy był rozwój procesu iteracyjnego projektowania inżynierskiego, który może być użyty do stworzenia niedrogiego systemu pomiarów zbliżeniowych do specyficznych zastosowań. Iteracyjny proces projektowania inżynierskiego został oparty na diagramach przejść między stanami. Jednopoziomowe diagramy przejść stanów mogą być zbyt skomplikowane, stąd w pracy zastosowano ich upraszczanie poprzez użycie kilku poziomów abstrakcji. Przedstawiono uzyskane wielopoziomowe diagramy stanów. Przedyskutowano jak użyć takie wielopoziomowe diagramy do zmodelowania procesu iteracyjnego projektowania inżynierskiego, na przykładzie tablicowego czujnika zbliżeniowego. W podsumowaniu scharakteryzowano uzyskane wyniki projektowe.

[1] Mikolaj Baszun, Warsaw University of Technology, Department of Electronics and Information Technology, Nowowiejska 15/19, 00-665 Warsaw, Tel. + 48 22 2347906, mbaszun@elka.pw.edu.pl
[2] Department of Mathematics and Computer Science, Fayetteville State University, Fayetteville, NC 28301, USA, bczejdo@uncfsu.edu

1. INTRODUCTION

There are many theoretical and practical solutions of ultrasonic proximity array sensors, available to assist the blind patient in recognizing and/or localizing the nearby objects [1,2,3,4,5]. Such systems (attached to the patient body or to his vehicle) could be used by the blind person himself or by the programmed system on his BPV. The monitoring data from such system could be also transferred by the automatic GSM link to the care centre Web database, together with another monitoring data transferred for the same patient [6,7,8,9]. The gap, however, still exists between expectations from such systems and the current capabilities. One of the constraints to introduce mass production for automatic driving systems is lack of inexpensive but reliable proximity sensors.

The objective of this paper was to develop, based on optimization methods [10,11,12], an iterative process of engineering design that can be used to produce inexpensive proximity measuring devices tailored to the specific applications. The iterative process of engineering design is based on state transition diagrams. Since one-level state transition diagrams can be quite complicated, we discuss how to simplify them by using many levels of abstraction. The resulting multi-level state diagrams are shown. We discuss how to use such multi-level diagrams to model iterative engineering design on the example of a proximity sensor.

The paper discusses a design process in which the iterative search for the final design is necessary. The iterative process of design can be supported by an appropriate software system. An overview of interactions between the engineer and a software system are given by [13]. Simple modelling techniques for such interactions are presented in [15,16]. Here, we present modelling details using a multi-level behavioural mode [14]. We carry out our discussions using iterative design of proximity sensor system.

The paper is organized as follows. First the discussions of iterative design are carried out for the design of a proximity sensor system (Section 2). Next, the one-level model of iterative design is presented (Section 3). The multi-level model is discussed in Section 4. Summary is given in Section 5.

2. DESIGN PROCESS ON THE EXAMPLE OF ARRAY OF PROXIMITY SENSORS

2.1. Proximity Sensor System

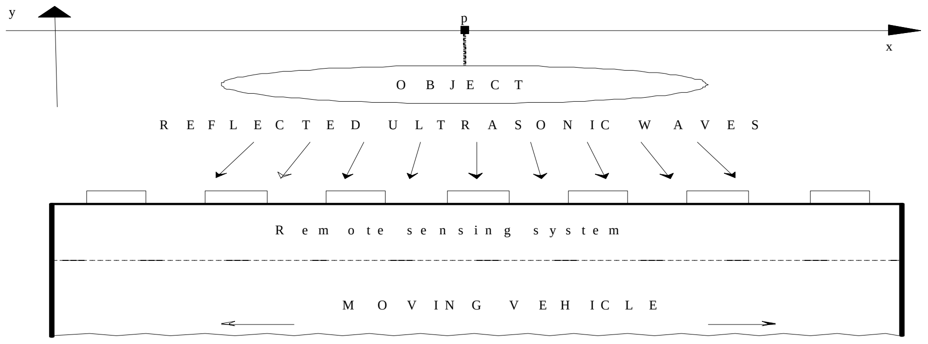
y
p
x
O B J E C T
R E F L E C T E D U L T R A S O N IC W A V E S
R e m o t e s e n s i n g s y s t e m
M O V IN G V E H IC L E

Fig.1. The vehicle sensor system interacting with the object.

Remote sensing systems continue for long time to be of great interest for many engineers and researchers. The applications of such systems for recognizing of remote objects could lead to new solutions, depending on their technical parameters. The example of simple kind of such system is shown in Figure 1.

System use ultrasonic waves reflected from recognized object and propagating to every sensor in multi-detector array connected to the specialized computer interface. The object is moving and needs to indentify the horizontal position of the object as is shown in Figure 1.

Figure 2 shows simplified geometry of the multi-detector array in relation to the object. The geometry is described by parameters N, p, d, l, r1, r2, ,rN, as shown in Figures 1 and 2.

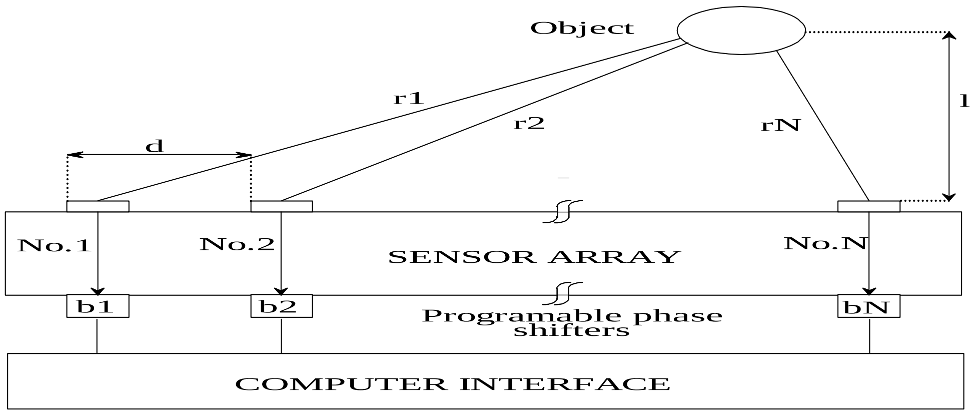

Fig.2. Configuration of multi-detector sensor system.

Additional parameters b1, b2, ..., bN shown in Figure 2 are phase shifts for electrical signals from detectors. The system measures the position p of the object by proper programmable computing of the additive signal S composed from N signals of the individual detectors according to the theoretical predictions. Resolution of such measurements depends strongly from the values of the geometrical and phase parameters shown in Figure 2. The designer needs to compute these values to obtain the required resolution of the measurement of the object position.

2.2. Iterative Design Process of the System

The system design process should result in finding values of parameters fully describing the system. Nature of these parameters, called design parameters (DP) can be quite different: material, geometrical, electrical, architectural, etc. Our example system is described by the geometrical DP: N, d, l, and by phase shifts DP: b1, b2, ..., bN. The N phase shift parameters we replace by one DP parameter b, thanks to our observations that the best case we have obtained for parabolic and symmetric distribution of the values b1-bN. So DP parameter b means coefficient at quadratic term at function describing distribution of b1-bN values.

So, for such case design process results in computing four values DP: N, l, d, b. However this case are related to very simple application of the designed system: measuring of one-dimensional displacement. Other solutions based on visual recognition or remote radar systems could be however more complicated as applied for object recognizing. Number of DP for such cases could exceed one hundred.

The design process should be preceded by the detailed analysis of physical phenomena outlining detailed properties of the system. Such analysis should lead to the creation of a physical model of the system. The physical model includes the constraints on the design parameters (C_DP). For the designed system such constraints C_DP are:

- d value must be greater than dimension of the individual detector in the array;
- l value must be limited to range resulting from technical realizability of the system;
- N value needs be limited by costs.

The results of constraints checking were stored as logical data CC_DP.

The application/market/utility requirements for the system product result in some constraints on chosen properties of the designed system. In this paper, for simplicity, we assume that only one such property of the designed system is considered. We will refer to such property as output characteristics (OC). Our OC is the dependence of output signal S on the position p characterizing the system ability to precise position measurement. The designed system exhibits the theoretical dependence shown as M(p) in Figure 3. During the design process shape of OC is tending step-by-step to the final M(P) shape.

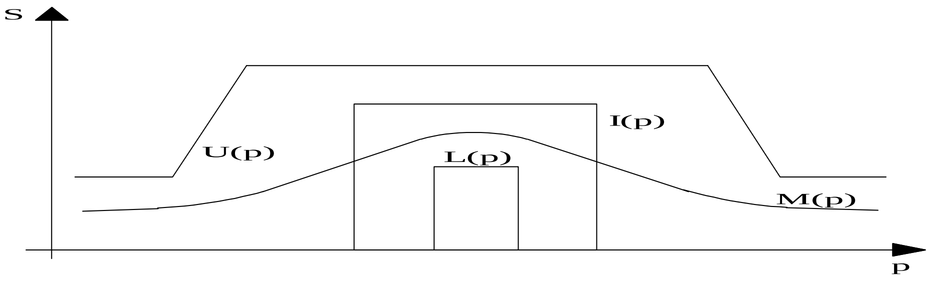

Fig.3. Output characteristics of the sensor system. S - output electric signal; p - object position; M(p) - final theoretical OC; U(p) - upper constraints to the OC; L(p) - lower constraints to the OC; I(p) - "ideal" output characteristics.

Constraints on output characteristics will be called C_OC. Our C_OC are shown as L(p) and U(P) in Figure 3. Such constraints at our system result from the requirements towards measuring resolution of the system (upper constraint) and from costs of extremely accurate system (lower constraint). The results of constraints checking were stored as logical data CC_OC. The criterion of evaluation of the quality of the output characteristics is based on comparison with the given ideal output characteristics (I_OC). Our I_OC is shown in Figure 3 as I(p). Such "ideal" output characteristics are given to designer as the best one wanted to obtain. To measure the closeness of these two characteristics an objective function (F_OF) will be used. This function we compute during every step of iterative design from expression: F_OF = integral (OC - I(P)).

The designer must be given or must develop a mathematical model of the system, giving complete mathematical relationships between all interesting quantities describing its internal and external behavior. The complete mathematical model also includes the precise algorithms of resolving of all mathematical formulas and equations describing the required relationships. Such mathematical model (F_OC) enables the computation of the output characteristics for any set of values of design parameters. Our F_OC was derived on the basis of field theory and circuit theory and is too complicated to present here.

It is a choice of the designer to use the proper design methods and accuracy. The choice affects the effectiveness of the design. Especially important may be planning of detailed strategy of searching for the final values of design parameters during design process. The design process could be treated as searching for final values of design parameters. Each parameter may have numerical value belonging to some set of allowable values. Very often such set includes all real numbers or real numbers restricted to a particular range. It means that the value of parameters may need to be searched from the infinite number of possible values. The only feasible approach is to use some kind of iterative search. In an iterative search a strategy of searching for the final design (A_DP) needs to be specified. A simple example of such strategy is a sequential search algorithm with the specification for changes of design parameters DP. The initial value of DP is specified first by the designer. In each cycle of the search the output characteristics and objective functions are computed. The process is terminated when the minimum value of the objective function is found. The search strategy should guarantee the global minimum of an objective function (F_OF) rather than its local minimum. Our design was based on iterative search. Search strategy was optional chosen by designer from various strategies offered by software. Values of the objective function F_OF are stored as all other values for several iteration steps. The design process was stopped after reaching of minimum value by objective function. This fact is stored by logical data M_OF.

3. ONE-LEVEL MODEL OF ITERATIVE DESIGN

A general model of an interactive design is shown in Figure 4. It consists of a Designer's Interface Module, and an Computations Control Module. The Designer's Interface Module allows the designer to provide initial values for design parameters and constraints on design parameters (setup request).

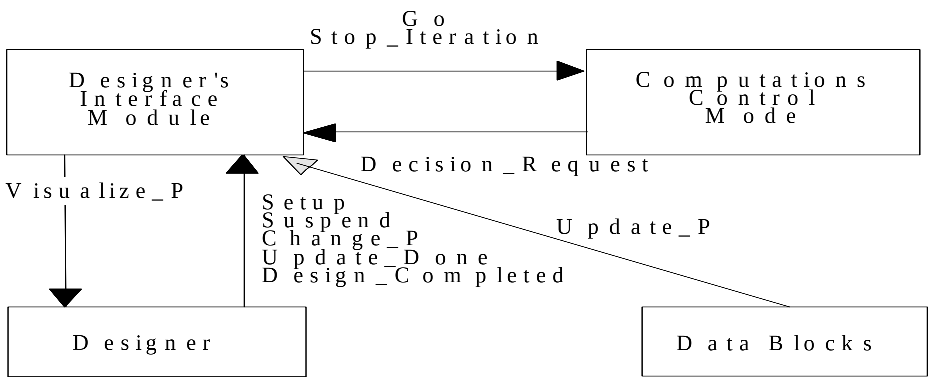

Fig. 4 A general Model of software for an design.

The initial formulas for output characteristics and constraints on output characteristics, the formulas for computation, and method of search and search increments of objective functions are provided by the designer also via Designer's Interface Module (setup request). Request ChangeP is to change these parameters and Update **Done** request informs the system that it can proceed with automatic iterations. The Suspend request stops the automatic computations and allows the designer to change the parameters again using ChangeP. The visualization of the design is available for the designer. The request DesignComplete is issued when the designer is satisfied with the design. The Computations Control Module receives requests Go and StopIterations from the Visualization Module. It can also send the request DecisionRequested when automatic computations can not proceed. Additionally, there are data blocks storing the information about the current state and the history of an design process.

Description of all visualization and processing modules will be done using a behavioural model [10]. This model is an extended state net that describes the dynamic behaviour of objects. It has three basic components: states, for each object in the given class, triggers that cause the transition of an object from one state to another and actions performed during the transition. Triggers can be either Boolean conditions or events. Each state is indicated by a rounded box and each transition is indicated by an arrow with a label. The first part of label (before slash) specifies the trigger and the part (after slash) specifies the action to be performed during transition. Boolean conditions are indicated by brackets while for events the brackets are not used. An action describes an activity that takes place during the transition.

The activity may consist of several steps, some of them can be generation of other events or operations on data blocks.

3.1. Designer's Interface Module

Initially, Designer's Interface Module (while in START DESIGN state) receives the request (the Setup event) from the designer to start the design. That results in starting the design process (the event Go is generated and sent to Internal Control Module).

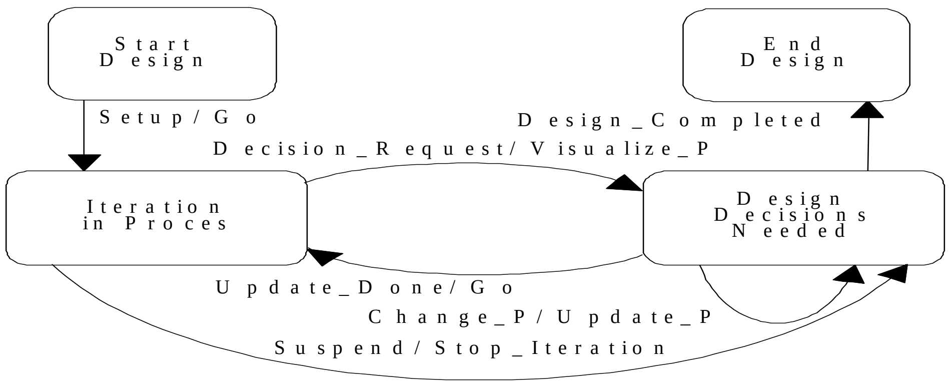

Fig. 5 Behavioral model for Visualization and Interactions Module.

The design process can be in two basic states corresponding to a passive (the ITERATIONS IN PROGRESS state) and active (the DESIGN DECISION NEEDED state) role of the designer. It is assumed that in the first state all iterations are done automatically and designer can only observe the current results. The designer can issue a request (Suspend event) in order to stop automatic iterations and participate actively in the next phase of the design. When this request is received all automatic iterations are stopped (event StopIterations is generated and sent to Internal Control Module) and all visualizations screens allow the designer to change appropriate design components. Another possibility for transition to the state corresponding to the active role of the designer is related with the situation in which a minimum of objective function is found or automatic iterations run into problem (the event DecisionRequest was received from Internal Control Module).

The second state corresponds to the active role of the designer (DESIGN DECISION NEEDED state). The designer can change all design parameters, constraints etc. using ChangeP request. Once all modifications are done the designer can issue a request (UpdateDone event) in order to start automatic iterations. More specifically, when this request is received the request to start automatic iterations is issued (event Go is generated and sent to Internal Control Module), and the state is changed to one corresponding to the designer passive role. If the designer, while in active mode, is satisfied with the design he can issue a request to end the design (the DesignComplete event causing transfer to the END DESIGN state).

3.2. Computations Control Module

Computations Control Module specifies the automatic design iterations. Initially, while in WAIT state, it receives a request to start the iterations (Go event). That causes a transition to a state (LOOP_BEGIN state) that enables a Computations Control Module specifies the automatic design iterations. Initially, (while in WAIT state), it receives a request to start the iterations (Go event). That causes a transition to a state (LOOP BEGIN state) that enables a sequence of operations defining one design iteration. This sequence includes:

1/ checking the constraints on design parameters (operation UpdateCC_DP(DP, C_DP) and transfer to CC_DP READY state); 2/ computing output characteristic (operation UpdateOC(DP, F_OC) and transfer to OC READY state); 3/ checking the constraints on output characteristics (operation UpdateCC_OC(OC, C_OC) and transfer to CC_OC READY state),- 4/ computing objective function (operation UpdateOF(OC, I_OC, F_OF) and transfer to OF READY state).

If, during the execution of this sequence of operations the constraints on design parameters are not satisfied (the condition CC_DP is false) or constraints on output characteristic are not satisfied (the condition CC_OC is false) then the iterations stop (the transition takes place to the WAIT state) and the appropriate message is sent to the other module (the event DecisionRequest). The automatic iterations also stop if the minimum of objective functions was obtained (the condition M_OF is true) or the request from the designer was received to stop iterations (event StopIterations).If, after finishing this sequence of operations there is neither request to stop iterations from the designer nor the minimum of objective function was obtained then the design parameter are modified according to search strategy (operation UpdateDP(A_DP)) and the next automatic iteration starts (transfer to LOOP BEGIN state).

4. MULTI-LEVEL MODEL OF ITERATIVE DESIGN

The one-level models can be quite complex. In order to simplify them we can use abstractions of state diagrams e.g we can create an abstract (high-level) Computations Control Module with some of the states replaced by an abstract state.

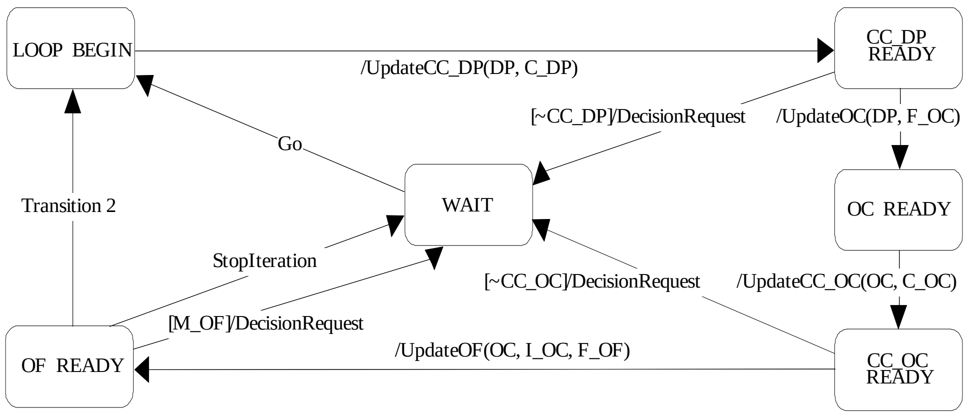

Fig. 6 Behavioural model for Computations Control Module. Transition 2 means: `StopIteration and `[M_OF].

4.1 High-level Computations Control Module

The Computations Control Module can be simplified by using two different levels of abstractions. In the first level that is shown in Figure 4 we have only two states WAIT and DO_LOOP.

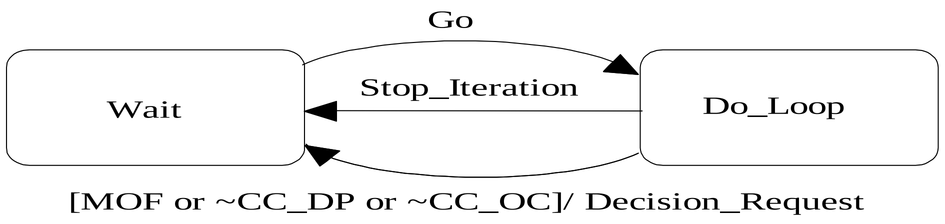

Fig. 7 High-level Computations Control Module.

Initially, (while in WAIT state), it receives a request to start the iterations (Go event). That causes a transition to a state (DO_LOOP state) that enables a sequence of operations defining design iterations. If,during the execution of this sequence of operations the constraints on design parameters are not satisfied (the condition CC_DP is false) or constraints on output characteristic are not satisfied (the condition CC_OC is false) or the minimum of objective functions was obtained (the condition M_OF is true) then the iterations stop (the transition takes place to the WAIT state) and the appropriate message is sent to EXTERNAL CONTROL MODULE (the event DecisionRequest). The automatic iterations also stop if the request from the designer (EXTERNAL CONTROL MODULE) was received to stop iterations (event StopIterations).

If, during the execution of this sequence of operations the constraints on design parameters are not satisfied (the condition CC_DP is false) or constraints on output characteristic are not satisfied (the condition CC_OC is false) then the iterations stop. The automatic iterations also stop if the minimum of objective functions was obtained (the condition M_OF is true). If, after finishing this sequence of operations there is neither request to stop iterations from the designer nor the minimum of objective function was obtained then the design parameter are modified according to search strategy (operation UpdateDP(A_DP)) and the next automatic iteration starts (transfer to LOOP BEGIN state).

4.2 Low-level Module describing DO_LOOP state

The state DO_LOOP in the previous Section 4.1 is an abstract state that can be defined more precisely. Initially, low-level Module describing DO_LOOP state is put in LOOP_BEGIN state. That enables the automatic design iterations specified by a sequence of operations similar to one in Section 3.2:

1/ checking the constraints on design parameters (operation UpdateCC_DP(DP, C_DP) and transfer to CC_DP READY state);

2/ computing output characteristic (operation UpdateOC(DP, F_OC) and transfer to OC READY state);

3/ checking the constraints on output characteristics (operation UpdateCC_OC(OC, C_OC) and transfer to CC_OC READY state),-

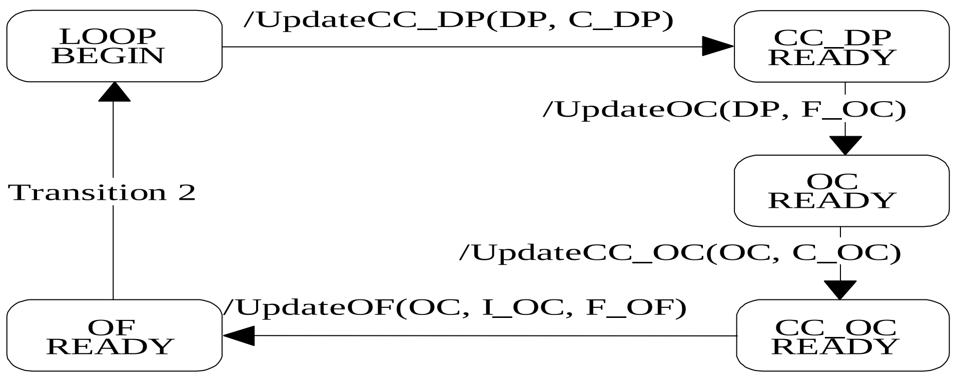

Fig. 8 Behavioral model for an abstract state DO_LOOP. Transition2 as in Figure 6.

4/ computing objective function (operation UpdateOF(OC, I_OC, F_OF) and transfer to OF READY state).

5/ checking if the minimum of objective function was not achieved (~M_OF), computing new design parameters (operation UpdateDP(A_DP) and transfer to LOOP BEGIN state).

If, during the execution of this sequence of operations one of the conditions is not satisfied then it takes place an exit from the low-level module to the high-level High-level Computations Control Module.

5. SUMMARY

In this paper we discussed modelling of the design process of array of proximity sensors. Such sensor systems have a potential of providing inexpensive solutions for vehicle control. The paper discusses a design process in which the iterative search for the final design is necessary. Simple modelling techniques for such interactions were presented before. Here, we have shown modelling details of proximity sensor systems. Our continuing experiences with multi-level models have shown that multi-level diagrams can provide a support for rapid design by using more efficiently a computation module.

As a application result we have obtained, for the assumed wave frequency value of 20 kHz, the theoretical data: /a/ object distance range 50 cm – 7 m; /b/ distance measurement resolution of 4 cm; /c/ angle towards the nearest object measurement resolution of 0.02 radian; /d/ curvature of the reflecting object calculating accuracy of range 10 cm (curvature diameter). The practical system, according to the obtained design data, is now under work.

6. REFERENCES

[1] Mattila P., Sirtola J., Suoranta R.: Two-dimensional object detection in air using ultrasonic transducer array and non-linear digital L-filter, Sensors and Actuators A: Physical, July 1996, Pages 107-113, Volume 55, Issues 2-3, 1996.
[2] Siciliano B., Sciavicco L., Villani L.: Robotics: modelling, planning and control, Springer Verlag Berlin, 2009.
[3] Fiorillo, A.: PVDF Based Sonar for a Remote Web System to Control Mobile Robots, Sensors & Transducers, February, 2010.
[4] Barshan B. Baskent D.: Comparison of two methods of surface profile extraction from multiple ultrasonic range measurements, Meas. Sci. Technol., 11 (2000), pp. 833–844.
[5] Pham D., Soroka A.: Ultrasonic distance scanning techniques for mobile robots, International Journal of Computer Aided Engineering and Technology, 2009 - Vol. 1, No.2, pp. 209 - 224.
[6] Baszun M., Czejdo B.: Transition to autonomous mode and remote control in vehicles with incapacitated driver, Logistyka, 6/2009.
[7] Baszun M., Czejdo B.: Real time medical advising in cyberspace and its security aspects, The VI International Conference Cyberspace2009, Masaryk University, Brno, Czech Republik, November 2009.
[8] Baszun M., Czejdo B.: An Interactive Medical Knowledge Assistant, volume “Visioning and Engineering the Knowledge Society”, Springer Verlag, ISBN 978-3-642-04753-4, 2009.

[9] Baszun M., Kruczyk M.: A real time telemedicine Web expert system for wireless monitoring of drivers/passengers, Monograph No. 122 (ISBN 1642-5278) edited by Politechnika Radomska, 2008.

[10] Yang X.: Introduction to Mathematical Optimization: From Linear Programming to Metaheuristics, Cambridge Int. Science Publishing, ISBN 1-904602-82-7, 2008.

[11] Mordechai A.: Nonlinear Programming: Analysis and Methods, Dover Publishing. ISBN 0-486-43227-0, 2003.

[12] Stephen Boyd S., Vandenberghe L.: Convex Optimization, Cambridge University Press, ISBN 0-521-83378-7, 2004.

[13] Baszun M., Czejdo B.: VSED - A Visual System for Engineering Design, Proceedings of The First World Conference on Integrated Design and Process Technology, Austin, 1995.

[14] Baszun M., Miescicki J., Czejdo B.: Multi-level Modeling of Iterative Engineering Design of Remote Robot System, Proceedings of The Second World Conference on Integrated Design and Process Technology, Austin, 1996.

[15] Baszun M., Miescicki J., and Czejdo B.: Modeling of Interactive Engineering Design, Proceedings of ESDA96 Conference, Montpellier, France, 1996.

[16] Rumbaugh J., Blaha M., Premerlani W., Eddy F., Lorensen W.: Object-Oriented Modelling and Design, Prentice Hall, New Jersey, 1991.

This work was partly supported by:

- Belk Foundation, North Carolina
- Funds on Science in 2007 – 2010 as Ordered Research Project of Polish Ministry of Science and Higher Educations.