Ziny Flikop

"Good enough" phenomenon and predefined bounded control

Abstract

The paper is an attempt to generalize an approach presented in [1, 2]. It discusses how a "good enough" phenomenon, honed by evolution, together with inherited and learned experience that all live creatures use in their struggle for survival, can be used for a decision-making process. Proposed approach is a variation of a bounded-input bounded-output (BIBO) method currently widely used for the system stability studies, and bounded-control (BC) used for the control of different systems. The paper proposes a method for a very fast defining a bounding surface as for the input as for the control subspaces. Enhanced "bounded-input bounded-output bounded-control" (BIBOBC) approach provides also adaptability to the bounded (soft) control and allows transferring controlled system along a suboptimal trajectory.

Key words: decision-making, bounded-input bounded-output bounded-control.

Introduction

Survivability of all live creatures, including humans, is based on two fundamental principles. The first principle is the ability of a creature to re-use an inherited, adopted, or previously learned experience (solutions) in different suddenly arising situations. It is a well known "If . . . Then . . ." approach. The second principle is that instead of relying on the "best" solution producing the "best" result for the precisely defined situation, a creature uses a non-precise solution and is satisfied with an outcome that is just "good enough". This flexibility allows a creature to use a limited set of solutions to cover about all its living situations.

Let us examine the second principle on the example of a human, because it is a less obvious one.

1. We select a "good enough" dress based on forecast, the type of activity we will be involved, and on our previous experience. This "good enough" dress can be used in spite of some variations in weather conditions and our activities.
2. We separate apples from oranges based of our knowledge of a "good enough" range in which apple parameters can vary. For example, it can be variations in the fruit shape, color, weight, skin texture, and so on.
3. A physician uses a database of symptoms and a history of diseases of multiple patients to find a "good enough" range in which health parameters can fluctuate and indicate, with some probability, a disease.
4. We recognize, with some probability, somebody's voice if its audio frequency spectrum is in our subjective "good enough" limits.

Some examples from psychology

5. Every human is unconsciously evaluating his/her well-being. This evaluation includes analysis of multiple directional oriented parameters, most of which have fuzzy values and have subjective importance. They are, for example, health related, financial; family related, represent living conditions, and so on. In addition, everybody has an unconscious fluid in time meaning to which limits these parameters can fluctuate and considered as "good enough." Depending on such evaluation, a person is either satisfied with his/her life or not. For example, a person can say either, "I am OK," or "I am not OK." In the case of a negative answer (solution), a person becomes motivated to start an activity to improve the situation, i.e., to move life parameters inside of "good enough" limits. If this activity results in improvement of the life parameters, then it creates positive effects on a person's psyche. It should be mentioned that "good enough" values are changing when values of evaluated parameters are changing, caused by either a person's activity or by some outside factors. If life conditions are changed and a person changes his/her activities to maintain "a good enough" estimate of his/her life, then the person demonstrates his/her adaptive abilities. (This remark is outside of the scope of our discussion, however, we have to mention, that if all parameters have "good enough" values for a long time, or if in spite of any activity person cannot improve values of the parameters that are important to him/her, then these situations can result in depression.)

6. When we ask for the advice, we unconsciously expect the advice to be in some "good enough" limits. If advice (solution) we received is within these limits, then there is a high probability that we will accept it. However, when the advice (solution) is outside of these limits, we will likely reject it. One can continue this list of similar examples.

Now, let us consider how this "good enough" approach can be transferred into a technical decision-making processes. First, we should define a controllable system (object, process, etc.) as a system that is determined by a set of input parameters, a set of output parameters, and some mechanism that allows mapping of valuable inputs into acceptable outputs. A system can be either linear or non-linear. In the case when some system parameters are fuzzy, for example, color, taste, etc., they can be defined via some digital directional sequences.

We can define a system state as a combination of system conditions, i.e., as a point into a multidimensional conditional input space. In the case of technical applications, since in the future we will use a **predefined** solution, instead of the term "solution", we will use the term "**predefined control**" or just "**control**". Some controls are relatively simple. For example, dress selection, separation of apples from oranges, recognition of somebody's voice, a patient's diagnosis, and so on. Some controls are more complicated. They include predefined controls (predefined solutions) as an initiator and a closed-loop control as an executor. For example, our reaction at a red traffic light. Some controls can be defined in the multidimensional control space. Controls can consist of predefined multi-steps with closed-loop inclusions. For example, a pilot's actions at take-offs and landings. Generally speaking, about all of our adult life our activities are based on complex initial predefined control (predefined solution) multi-steps with closed-loop inclusions.

For the analysis we can use a real object or either its physical or mathematical model. In some cases, human expert participation is necessary. In the analysis below, we will assume that we have a mathematical model of an object and specialized software.

In the very beginning, for a particular situation (let us name it an "origin" point in the input space) we can find via optimization a "best" possible control that maps the "origin" into the "best" desired output (in the one-dimensional case see Fig. 1.) If for every possible input we will define the "best" control that provides us with the "best" output, then we finish with an infinite number of controls (in the one-dimensional case, see Fig. 2).

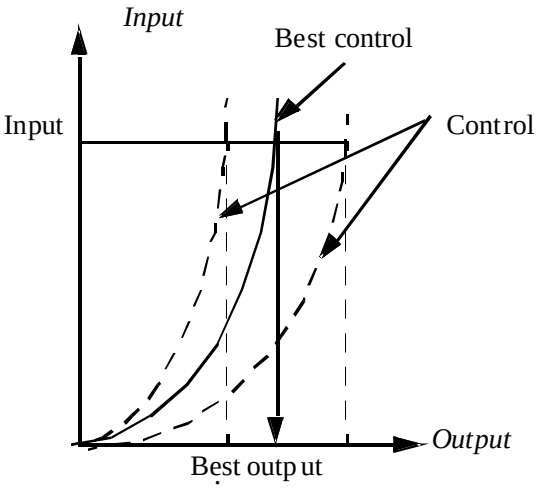

Fig 1.

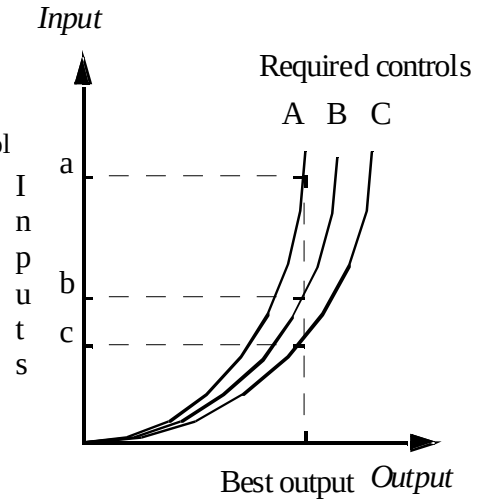

Fig 2.

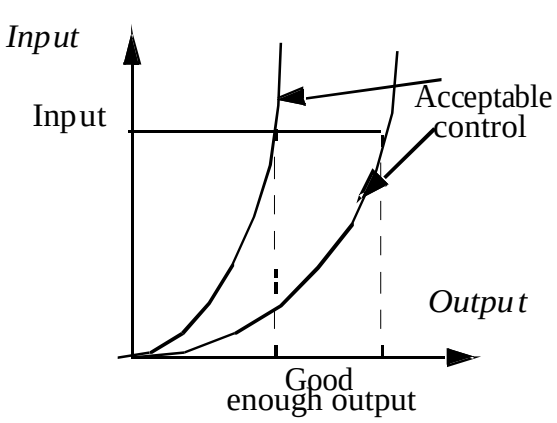

Fig 3.

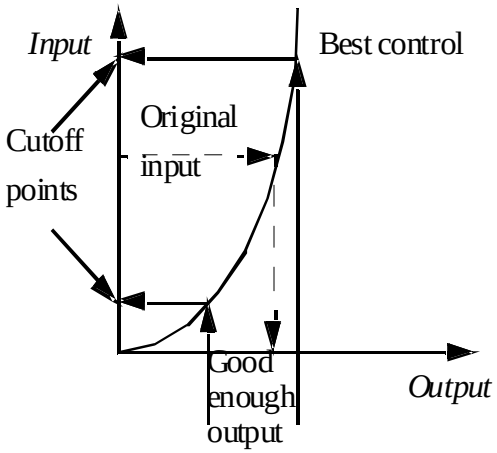

Fig. 4

\

However, as system designers, we may define limits in which system output can fluctuate and, in spite of these fluctuations, we can accept it as "good enough". If we define a "good enough" bounded output area, then we can have two extreme situations. Namely, for one input we will have a set of "acceptable" controls (in the one-dimensional case, see Fig 3), or for one "best" control we will have an acceptable input area (in the one-dimensional case, see Fig 4).

It is possible to have an intermediate solution, when instead of one "best" control; we can find a set of controls that satisfies "good enough" bounded output conditions. However, the price we pay for this is a shrinking input area (in the one-dimensional case, see Fig 5).

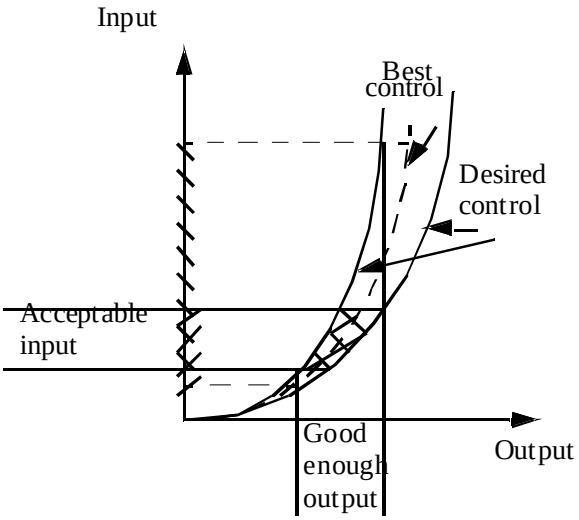

Fig 5.

With a fixed "best" control, the more input deviates from the "origin" point, the more output departs from its "best" value. It means that with too-wide deviations of input a "goodness" of a found "best" control can deteriorate to the extent that departed input will be mapped outside of a "good enough" area, i.e., control becomes either useless or, in the worst case, counterproductive. Border points in the input space that the "best" control cannot map into a "good enough" bounded output are the cutoff points for this control (in the one-dimensional case, see Fig 4). In the multidimensional case, cutoff points create an outer surface of a subspace in the input space (bounded input). The "best" control is capable of mapping any point in this subspace into a "good enough" bounded output.

Like in real life, usage of a "good enough" approach for technical applications consists of two phases: learning and execution. Let us start with an analysis of the learning phase in more detail. We will do our analysis of a generic system that is described by an input, output, and a control, which can all be multidimensional. It is obvious that dimensionality of input space, control space, and output space can be different.

Learning

1. As a first step, for the given input represented by the "origin" point in the multidimensional input space, we will find via optimization a "best" control that maps input into the "best" desired output. This multidimensional optimization process is complicated; however, the result of it is similar to the case shown in Fig. 1. Then, if a "good enough" bounded output space that includes the "best" desired output is already defined, we can identify via reverse mapping a bounded input subspace, which "best" control maps into bounded output. We will do the reverse mapping in few steps:

2. In the very beginning, we normalize the input space at the "origin" point by defining a measure in some units for each parameter (dimension). For example, a unit of temperature can be 1/100 of a maximal considered temperature; the taste of a fruit can be 1/100 of maximal taste units, and so on.

3. This normalization allows us to define a directional ray, which originates at the "origin" point and is going in the input space in a random direction. We can achieve this by multiplying each dimensional unit by corresponding to it random number.
4. Now, we can start to move the input away from the "origin" along a randomly selected ray. As a result, the quality of the output will steadily diminish until we arrive at a border of "good enough" bounded output. The input that the "best" control maps into this border is a **cutoff** input point. Considering that a ray is one-dimensional, we can use a very fast one-dimensional optimization algorithm to find a cutoff point for the combination: direction, "best" control and a "good enough" bounded output. Coordinates of a found cutoff point are memorized.
5. By using another set of random numbers, we can define a new directional ray that originates in the same "origin", repeat the optimization procedure and memorize a new cutoff point.
6. After we repeat step 5 a number of times, we will have a set of cutoff points that determine a boundary of the input subspace, any point of which can be mapped by the "best" control inside of a "good enough" bounded output.
7. The cutoff's points allow us preliminarily define by some polynomial a multidimensional surface that bounds the found input subspace. We consider that, with exception of systems designed by man (for example, the Stock Exchange), this surface is convex. In addition, the input subspace restrained by this surface does not contain sub-spaces input points in which cannot be mapped by analyzed control into a "good enough" bounded output. We rely on these hypotheses until otherwise proven. Polynomial description of the surface can be done by using a "Fit" function of [3], or another fitting algorithm. The result of a fitting provides us with information about the polynomial and the value of the least square error corresponding to it. Using this polynomial allows us to speed up process, since now we can continue selection of testing input points that are very close to the bounding surface.

These points can be randomly selected one-by-one and each point becomes the "origin" for the procedure described in steps 3 and 4. Each additional cutoff point is used to correct the polynomial and recalculate the value of a new least square error. We continue this procedure until this error is stabilized. Then we terminate it and consider that the polynomial is found. For the two-dimensional input, this procedure is illustrated by Fig. 6. After the polynomial is found, we memorize a trio: a **polynomial** that defined the bounded input subspace, the **"best" control** that maps any input of this subspace into a "good enough" bounded output, and the "**good enough" bounded output** itself.

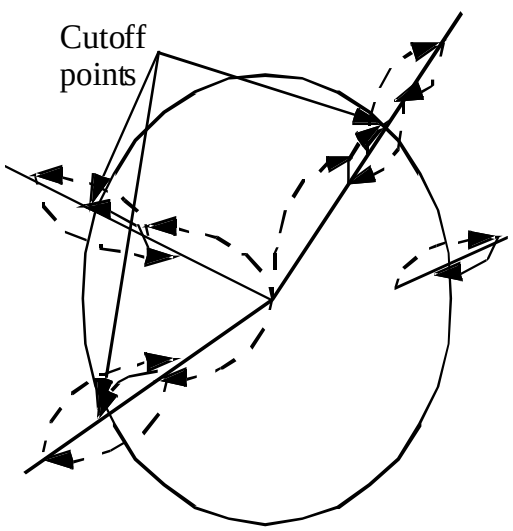

Fig 6.

The approach described in steps 1-7 has been successfully verified on the model of a telecommunications network [1, 2]. This model

represents a system with 31-dimensional input, 43-dimensional control and one-dimensional output. For the polynomial fitting was used a “Fit” function from [3].

8. Now, we can try to put some flexibility in the already found “best” control. To do so, we can, for the same “best” output and “good enough” bounded output, chose an input point located inside of a bounded input subspace defined by the polynomial. We will consider it as a new “origin” point. If we repeat steps 3 through 7, we can find another “best” control and another bounded input subspace that is intersecting with the previously found input subspace. As a result, all input points that belong to the intersection will be mapped into the “good enough” bounded output by either of the controls. We can repeat step 8 a number of times and define such control subspace that any control in it will map any input point belonging to the intersected input subspace into a “good enough” bounded output.
9. We also can repeat steps 1-7 for the same “best” output and “good enough” bounded output, however now we will choose input points that are outside of intersected input subspace, however are in the close proximity to it. If bounded input subspaces for these points are intersecting with the previously found intersection of input subspaces, then we can accept the newly found controls as valid controls for the “good enough” bounded output. This will allow us expand a control subspace found in 8.
10. We can repeat steps 8-9 as many times as our time resources allow. As a result, we will have an intersected input subspace all points of which are mapped into “good enough” bounded output by any control belonging to the found control subspace. We hypothesize that this control subspace is convex and try to determine its bounding surface by some polynomial. We define a **bounded input subspace** as a not empty intersection of input subspaces and a set of permitted controls via a **bounded control subspace**.

Now, we can modify our memorized trio. Since any control that belongs to the bounded control subspace maps any input point of the intersected input subspace into the “good enough” bounded output, we memorize a trio as follows: **bounded-input bounded-control bounded-output solution.**

Steps 1-10 can be repeated for completely different states of the analyzed system (points in the input space). As a result, we decompose an input space on controllable subspaces and will have a library of solutions for different operational situations. A set of memorized trios allows us to execute a system control in real time.

In addition, we can maintain the same “origin” point, but move the “best” output and “good enough” bounded output along some path. Multiple repetitions of steps 1, 3-10 will allow transfer the analyzed system from one state to the other state along a “good enough” trajectory.

If we maintain the same “best” output and “good enough” bounded output, but move the “origin” point in some direction, then the result of steps 1, 3-10 will let us equip analyzed systems with adaptive abilities.

We can reduce an educational time by utilizing an available expertise. For example, we can mimic educational processes that humans and some animals use in transferring their own expertise to the children. In complex cases, experts can teach, for example, a robot to do different tasks. The experience of a test pilot can be used for the development of a recovery procedure in case of emergencies.

Possible applications of the proposed methodology

1. It becomes feasible to create fail-proof systems. To do so, we can limit system control by permitted (bounded) control space and, as a result, prevents some accidents. This approach, for example, allows prevention of airplane crashes, similar to the crash of the Egyptian airliner, or the Chernobyl accident.
2. We can prepare in advance a set of flexible solutions for a robot. The robot will execute these solutions depending on the required task and environmental conditions. In addition, a robot can adjust its reactions upon changing environmental conditions or output requirements. This is a very important ability in the cases when robot works independently without human intervention.
3. It is possible to develop in advance a set of recovery procedures for natural disasters.
4. It is a very well known that precise plans developed in advance become useless in real situations. Applying proposed methodology for the development of adaptable plans can be very productive.
5. Systems (objects), for example engines, can be transferred from one state to another via a sub-optimal trajectory.
6. The experience of a test pilot can be used for the development of recovery procedures in case of emergencies.
7. Time for the development of systems with bounded-input bonded-control bounded-stability-output can be drastically shortened.
With some imagination, one can significantly expand this list of possible applications.

Execution (Predefined Control)

Execution of the proposed methodology is relatively straightforward. Control system monitors input conditions of the controlled system and defines to which input subspace the current input belongs. If input belongs to the subspace defined during the learning process, then control corresponding to this subspace is executed.

In the cases of recognition, information about object (sound) parameters is used to define an object via a library of solutions. A found solution gives us a probability that recognition is correct.

References

1. Z.Flikop, “Input set decomposition and open-loop control in telecommunications networks”, Proceedings of the American Control Conference, Seattle, 1995.
2. Z.Flikop, “Some problems with the design of self-learning open-loop control systems,” European Journal of Operational Research, Vol. 81, 1995.
3. S. Wolfram, “Mathematica,” Second Edition, Addison-Wealey, Redwood City, CA, 1991.