9/7/14

Saved in Documents\Papers\Homily

White Paper

A Short Essay on the Suitability of Implemented Turing Machines as Mission Controllers for Military Drones and Autonomous Vehicles

Robert B. McGhee (robertbmcghee@gmail.com)

Naval Postgraduate School, Monterey, CA 93950

1. Introduction

This is a short essay written by me to colleagues who I think may be interested in both theoretical modeling of robot mission control by Turing Machines (TM), and in perhaps in actual application of such models. I have decided to write this communication just to get some ideas off my chest, and also because I have heard that DARPA is now showing a renewed interest in approaches to autonomous vehicle mission specification and control. To give substance to what follows, I have attached three recent publications by myself and my collaborators at the Naval Postgraduate School (NPS). Please feel free to share this document and/or the attachments with anyone you think might be interested.

To my chagrin, while teaching Artificial Intelligence courses at NPS, I found that TM are nearly always treated as purely theoretical models for computation, and are examined by mental processes only. I don't believe that this is the best way to deal with this foundational topic. Instead, in the past, I have provided my students with a general TM simulator and required them to encode and test the finite state machine (FSM) part of a TM to solve a particular problem [1]. In so doing, I came to the (evident) realization that testing an *implemented* TM (in contrast to the usual *theoretical* TM) requires a human *external agent* to load the FSM state table and a data tape, and to then interact with the host computer to obtain and analyze specific results. In a similar way, if one wishes to use an FSM to describe and control a mission for a robot, then the robot becomes an external agent, and for parts of the mission may entirely replace a human *remote operator* [2]. In our work at NPS, we have decided to call a generalized implemented TM, furnished with both a robot and a human operator, a *mission execution automaton* (MEA). We have also defined and implemented a *universal* MEA (by analogy to a universal TM), as a *mission execution engine* (MEE) [2].

The particular MEE we have implemented at NPS [2] is encoded in *Prolog*. At first, in earlier work, we (at NPS) thought that the ability of Prolog to perform backward chaining inferencing was essential to mission execution. Particularly appealing to us was the fact that mission orders written in Prolog closely resemble orders written in English for human execution. However, we have changed our

minds. With more experience, we have come to believe that all such missions can be expressed as *flow charts* [3], and that such charts are at a higher level of abstraction than Prolog code. Moreover, we further believe that flow charts are understandable by almost anyone experienced with writing or executing mission orders (or procedure manuals) intended for human execution. Thus, we would now prefer to use a graphical programming language, such as *Simulink*, to autogenerate executable code. We have not yet done so. Furthermore, to our surprise, we have not to date found any published work that takes this step.

To make the above remarks more familiar, there are many application programs that automate the *drawing* of flowcharts intended to define processes to be executed by human beings. Yet, so far as we have been able to determine, none of these autogenerate interactive code for human execution as we have done in Prolog in [2]. If any of you know of such a system, I would be pleased to hear from you.

Frankly, I am amazed by the state of affairs I have described above. It now seems to me that millions of people sitting in cubicles are acting as external agents under the control of a large MEA. I know of no formalization of such complex systems as some sort of extended TM as I have outlined above and in the references attached. Moreover, I would think that there would be considerable commercial value in a code package that would *animate* flow charts to make them *active* rather than *passive*. If you are aware of any such software, please let me hear from you.

2. Application to Drones and Mission Simulators

The above discussion considers only automation of mission control for human execution. In the case of *drone*s (understood to mean any unmanned vehicle with a remote operator), an interface between the human controller and the vehicle is required to allow the human to send executable commands to the vehicle, and to process responses from the vehicle. In defining the *rational behavior model* (RBM) software architecture [3], we have elected to call this interface the *tactical* level, and the mission control level (the TM finite state machine) the *strategic* level. The real-time control software residing on the remote vehicle is referred to as the *execution* level.

If a computer-based mission simulator is available, and if the mission is controlled by a MEA as described above, then two forms of mission order testing are possible. First of all, if the mission FSM (i.e., the MEA) is *loop free*, then exhaustive testing of all possible dialogs between the human tactical level and the MEA is possible. In [2], this is illustrated for an example submarine target search mission. When possible, this provides the strongest possible kind of proof of *mission correctness*. On the other hand, when loops are present in the flow graph representation of the mission orders (and therefore in the associated MEA), exhaustive pre-mission testing must, in general, be limited to interactive verification of all of the state transitions and outputs of each of the MEA states. This case is also illustrated in [2]. It is our opinion that either of these outcomes is better that the usual approach of relying entirely on human judgment and experience to establish the correctness of mission orders.

A further benefit of MEA mission control is that a requirement for run-time verification of the ethical correctness of every order to the tactical level can be incorporated into the MEA. Of course, in the case of drone control, this evaluation is required as mission events unfold, and depends therefore on the expertise and training of the remote operator tasked with executing tactical level vehicle control. References [2] and [3] provide an example of such *ethical branching* during mission execution.

It is the opinion of the authors of the attached references that the advantages of controlling drones using MEA's with built-in ethics checking at run time are difficult to understand without examining some detailed examples. Therefore, if I have captured your interest at this point, please try to read the attached references. Of course I will be happy to reply if you care to contact me by email to clarify any of the ideas or results in these documents.

3. Application to Autonomous Vehicles

Evidently, applying RBM to autonomous vehicles requires that the tactical level be encoded to automatically accomplish the translation from strategic level commands to execution level commands, and to integrate responses from the execution level to answer strategic level queries. A very serious question is: is this even possible? Moreover, if possible, must the automated tactical level retain all of the capabilities of a human remote operator? Our answer is that we believe that RBM works well with autonomous vehicles and that human level sensing and reasoning is not required to obtain effective systems. This means that, in our opinion, artificial intelligence (AI) is not required to obtain a primitive (but useful) form of ethical behavior in an autonomous vehicle [3]. This is certainly not a generally accepted view. Moreover, even though we demonstrated some of our claims regarding RBM software in at sea experiments with unmanned submarines beginning over 20 years ago, and have shown a software methodology to smoothly transition from drone operation to autonomous operation [2], much remains to be done to establish the practical value of what we are advocating. More experience with the use of RBM in a variety of vehicles is especially needed, If you or anyone you know is interested, I invite your participation in such an ongoing effort, and welcome further discussion about the ideas presented here and in the attached documents.

REFERENCES

1. McGhee, R.B., Brutzman, D.P., and Davis, D.T., *A Taxonomy of Turing Machines and Mission Execution Automata with Lisp/Prolog Implementation,* Technical Report NPS-MOVES- 2011-2, Naval Postgraduate School, Monterey, CA 93943, April, 2011 (attached).

2. McGhee, R.B., Brutzman, D.P., and Davis, D.T., *Recursive Goal Refinement and Iterative Task Abstraction for Top-Level Control of Autonomous Mobile Robots by Mission Execution Automata - A UUV Example,* Technical Report NPS-MV-12-001, Naval Postgraduate School, Monterey, CA 93943, March 2012 (attached).

3. Brutzman, D.P., Davis, D.T., Lucas, J.R., and McGhee, R.B., “Run-Time Ethics Checking For Autonomous Unmanned Vehicles: Developing a Practical Approach”, *Proc. 18th Int’l. Symp. On Unmanned Untethered Submersible Technology,* Portsmouth, NH, Aug. 2013 (attached).