Publishing papers and books for autonomous vehicle agents ⋆

S M Veres * **and L Molnar** *

* *School of Engineering Sciences, University of Southampton, Highfield, Southampton, SO17 1BJ, UK. Email: s.m.veres@soton.ac.uk.*

Abstract: The paper presents an information processing system for autonomously operating vehicles where engineers can write ”publications” written in English that the autonomous systems can ”read” to acquire knowledge such as various skills and behaviour policies. Knowledge about how to perform feedback control based operations, how to do dynamical modelling, path planning, servo and tracking control skills, vision based feedback control, etc. can also be transferred. The ”publications” are similar to engineering booklets with contents, sections, subsections in English, that can appear in HTML, LaTex(pdf) formats. The same paper's HTML file can be read by an agent on board the autonomous vehicle and after reading the paper the agent knows how to alter its control of the vehicle. There is no need for an application engineer to read a journal publication and implement it by programming, the research engineer's methodological work is directly utilized by the autonomous vehicle or robot. Engineers different from the author of the ”publication” can also read the papers and learn the details of how the vehicle operates, how decisions are reached and how skills are performed. Not only will users of the autonomous vehicle clearly understand how it operates but will also know its limitations to avoid misuse or misunderstanding. Users can modify English sentences in the ”publication” to modify the vehicle's behaviour or how its skills are performed.

Keywords: Autonomous vehicles, world modelling, map interpretation, collision avoidance, natural language programming, artificial intelligence, publishing knowledge for autonomous systems.

1. INTRODUCTION

The process of knowledge distribution among scientist and engineers is by way of publishing at conferences, in journals and books. They describe in technical papers how a control method works, how a navigation method proposed by an author works. Also information processing and decision making procedures are published by research engineers (REs). Other engineers, let's call them application developer engineers (ADEs), read these technical papers, conceptualize the meanings and may opt to implement the methods in practice. There are thousands of control and signal processing engineering papers written every year worldwide. Their impact on industry is patchy as many of the papers are not read or chosen to be ignored due to financial necessities of companies. Knowledge transfer to industry (KTI) is slow as is well known. Cynics may even ask: ”why do we need to speed up KTI?... we are fine as we are”. Others may say: ”this is a topic for our knowledge engineers ...but artificial intelligence is far from being able to help yet.”

Is it possible today to make a machine to ”read” a technical book written in English sentences to formulate methodologies a the conceptual level? Some would say: ”natural language interfaces” have been developed and you can use various controlled English languages”. These need a priori knowledge and definition of meanings means and you need to know the grammar .... ”the problem is a complexity nightmare”.

And yet this paper is describing an existing system whereby engineers can write documents in English that can also contain equations, figures and images, quotes, numbers, physical quantities, etc. that suitable intelligent agents can read and apply it in their daily work. This paper describes a system with the following practical features:

(1) The authors and readers of these machine readable documents (called ”system English” papers, for short called sEnglish papers) do not need to learn grammar, just to apply common sense. The most important thing for an author is to have conceptual clarity in their area and express that in a suitable style: ”writing skills are needed”. It only takes a few hours to be trained as an author.
(2) The autonomous systems can read these special engineering publications in sEnglish. sEnglish publications are of special format with contents, sections and subsections. They may contain images, mathematical formulae to support understanding both to humans and to autonomous systems who read them.
(3) The autonomous systems who read these documents do not need access to a central dictionary or library of meanings. There is no need for building up an infrastructure so that autonomous systems can read sEnglish books. The system can be used immediately, without any investment. Autonomous vehicles (AUVs, UAVs, AGVs and spacecraft), robots, toys can be built that understand natural language documents without a supporting infrastructure to be set up first.

The question could be asked that: ”do we need publications for autonomous systems that humans can also understand”.

⋆ S M Veres is Professor of Autonomous Control Systems at the University of Southampton, UK. This project was supported by hardware and software provided by SysBrain Ltd, 3 More London Riverside, London, SE1 2RE, UK.

The right question to ask is rather ”do intelligent autonomous systems need to read publications that humans can also read”. We argue in this paper that the answer is an emphatic yes. This is not just a ”nice feature”. Machines reading publications is going to become a necessity if we want truly intelligent autonomous systems in the future.

The reason for ”publication for autonomous systems” is needed is complex and the two most important factors are:

(1) We do not only need to pass on knowledge to our ”servant robots” - we need to be aware what they know and how they know it, as that can help to avoid misunderstanding.
(2) The development of mental and physical skills of intelligent autonomous systems (by humans) is much faster than how fast we can economically afford a capable hardware to be replaced. Hence we need ”software upgrades” - but how to do it? We argue that software downloads for intelligent autonomous systems are old-fashioned and outdated. These autonomous systems should read books on mental and physical skills and facts about the world from documents that their user can also read with ease and hence they can have a shared understanding of a selected part of the world.

In this publication system the manufacturer of a vehicle platform or robot can invite publications from research engineers to enrich the knowledge base of agents controlling these platforms in a format that also merits current standards of scientific and engineering publications. This could for instance be the case for the use of autonomous vehicles, robotic pets and intelligent toys, gardening robots or agricultural unmanned autonomous aerial vehicles.

Publications for autonomous systems is a logical next step of information systems in a historical perspective [Veres]:

(a) There was first verbal communication between people.
(b) Writing was developed to record events and knowledge.
(c) Later book was invented, i.e. multiple copies of written material were printed that masses of people could read.
(d) Journal and newspaper publishing and the WWW exists to distribute knowledge.

In the computer science community the idea of ”natural language programming” has been largely considered impossible and impractical due to ambiguity. This paper reports about a complete system for autonomous systems unambiguously understanding human readable books to enhance their skills and knowledge about the world.

There is an sEnglish (short for system English) authoring Tool (sEAT) and sEnglish reader agent system (sERA) [Sb] that does the job described above. A simple reactive agent (sERA) that ”understands” these papers and can run on embedded computers or on a PC [Sb] is available. The system is well tested and has been formally verified [Sb]. This paper describes this system of ”publications for autonomous systems”. The methodology is illustrated on the control of an experimental autonomous ground vehicle shown in Fig. 1.

2. OPERATIONAL PRINCIPLES

The process of publishing for autonomous systems can be split into three parts: (1) authoring papers (2) distributing them in HTML format for web browsers and as PDF documents (3) making agents to read these papers and let them use their content in their operations.

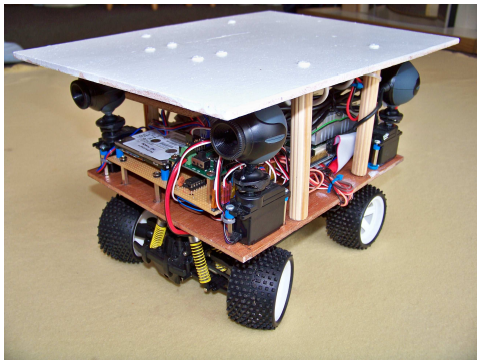

Fig. 1. The autonomous ground vehicle platform used for the demonstration.

2.1 Authoring papers

The process of publishing for both agents and fellow engineers, is displayed in Fig. 2. There is an authoring tool to write a paper in English sentences where the meaning of sentences is explained by other sentences until the meaning reaches signal processing level and no more conceptual meanings need to be defined: at this level MATLAB code or similar high level code (Octave, SciLab or Phyton) is inserted that represents ”unconscientious” operations. Goals, actions and modelling statements about the world can all be expressed in natural language sentences. Fig. 3 displays a window of the authoring tool sEAT. The start of an example is presented in Fig. 4 as a web document in HTML. The high level code can be interpreted by the agent for its realtime system (e.g. TTTech or LonWorks)

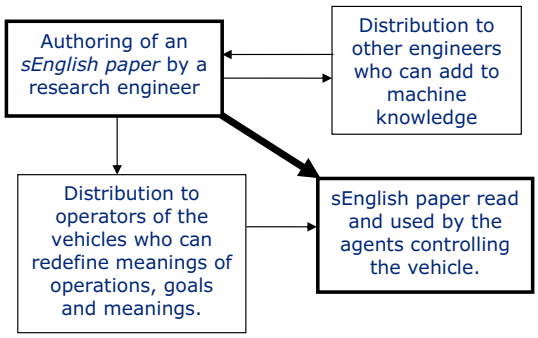

Fig. 2. The principle of authoring and distributing natural language programs that are presented as sEnglish papers.

The abstract of the paper describes the conceptual relations to other areas of knowledge and where the contribution of the paper lies and in particular it refers to the type of hardware system where the agent operates.

An sEnglish paper starts with an ”Abstract”, ”Conceptual structures” used and continues with sections and subsections that describe the meanings of activities. An activity can be represented by several sentence formats meaning the same. The actual meaning of a sentence can be always explained by other sentences that may be explained by further sentences or they refer to a piece of computer code.

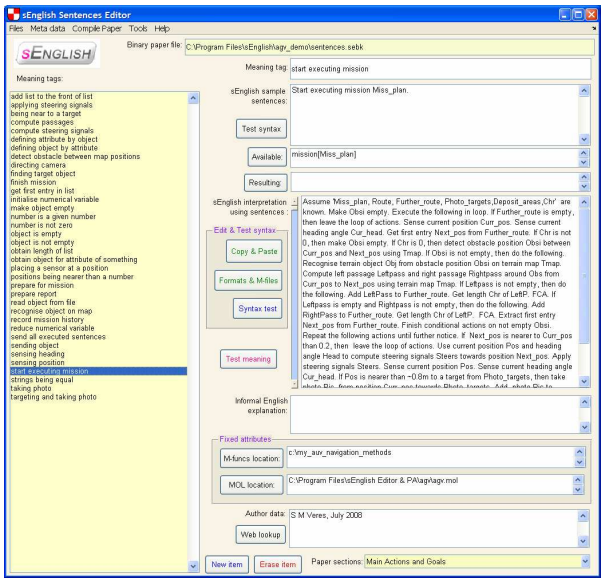

Fig. 3. One of the authoring tool GUIs to type in meanings of sentences by other sentences.

3. DISTRIBUTION OF PAPERS

sEnglish papers can be either in (1) HTML formats for direct transfer to agents through a local network or the Internet or in (2) PDF formats for mainly human reading of the papers. Figures 4 and 5 show the beginning of an sEnglish paper in HTML. The contents of the paper in Fig. 5 is not for human consumption only, the agent reads the concepts and activities from the contents and the rest of the text of the paper.

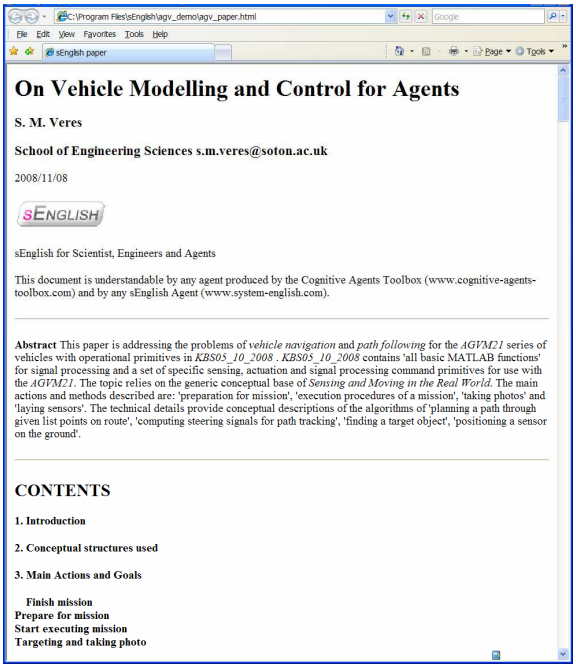

Fig. 4. The initial part of a natural language program (NLP) written in sEnglish and represented as a web page (HTML document).

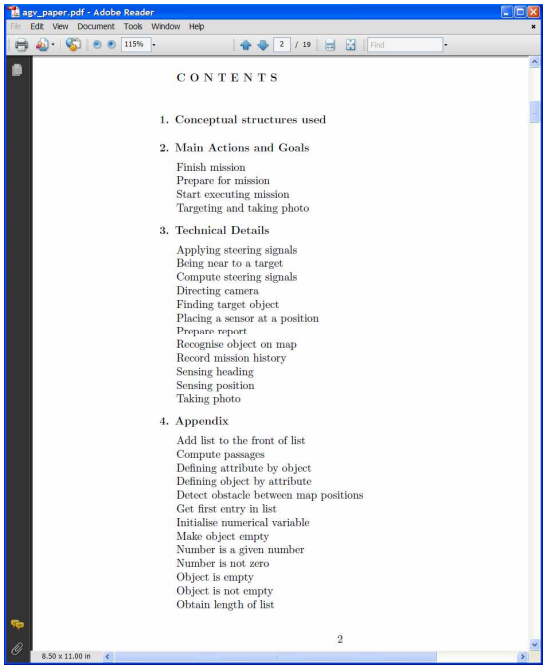

Fig. 5. The contents part of the same natural language program (NLP) written in sEnglish and represented as a PSF document (HTML document).

3.1 sEnglish based reactive "behavioural" agent

Reactive agents are simple and can be based on finite state machine definitions. There are also layered agent architectures that evolve abstractions from low level sensors to symbolic computation and then from decision making back to coordination and planning in detail to low level.

A simple agent that has finite-state-machine transitions prescribed by exit conditions and entry conditions (such as those defined in hybrid systems or in StateFlow^{TM}), can be defined by the use of sEnglish sentences [Molnar]. This agent is called reactive or situated as the environment and internal events determine the course of action the agent takes. There is no deliberation of intentions and plan selection or plan building. Reference [Molnar] provides a methodology of how reactive agents can be defined using an sEnglish document. The agent defined in sEnglish can be compiled into StateFlow^{TM} for simulation and into ISPL (interpreted system programming language) for formal verification by model checking.

Another simple agent, that can execute reactive behaviour, as defined in an sEnglish paper provided to the agent, is available under the name sEnglish Reader Agent (sERA) in association with the sEnglish Authoring Tool [Sb] . This agent can read sEnglish publications in HTML format and use them to control an embedded system or a PC based control system. Its user interface is displayed in Fig. 6. sERA's main functionality includes:

- Read an sEnglish paper in HTML file from a computer memory device, from a local network location or from a URL as instructed via the GUI of the agent.

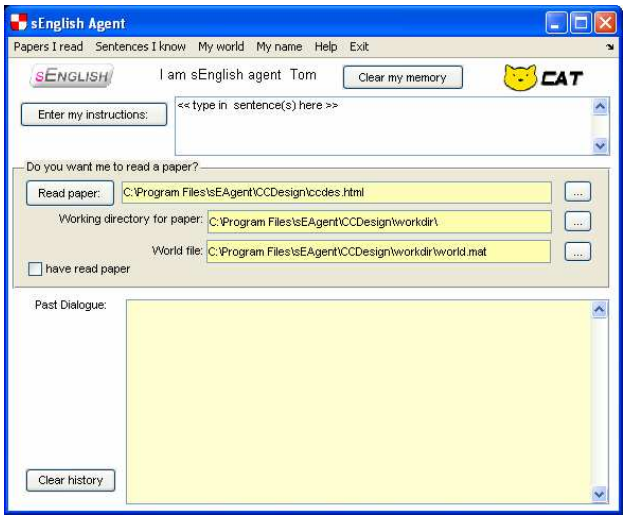

Fig. 6. The user interface of the agent that can read and interpret technical papers in sEnglish to carry out tasks and achieve goals. Sentences used can be used for both execution as well as for communication.

- Execute the meaning of a sentence entered via the GUI dialogue field of the agent
- Receive a message over the network that asks the agent to read an sEnglish paper from a depository on the Internet.
- Execute the meaning of a sentence send to it via network such as "Start mission M01.".
- Display history of a dialogue between itself and other agents, including any human contact and communications.
- Perform messaging, sensing and control operations as prescribed by sentences with meanings as defined in the sEnglish paper read by them.

Though simple, sERA based agents easily lend themselves to formal verification of behaviour. The sEnglish sentences define abstractions of actions, environmental events and sensing/modelling of the environment. For easy understanding of operations for human operators, the state transitions can also formulated by "If... , then ... ." sentences.

3.2 Advanced, deliberative BDI agents

Belief-desire-intention (BDI) agents can be made capable to read and use sEnglish papers with small (re-)configuration effort. We illustrate this on the Java based *Jason* agents.

Agent based control of autonomous vehicles or robots is the only proper way of writing software. Assume someone ignores this statement and writes a code for a vehicle or robot using a hybrid system specification as a finite state machine with continuous flows within states (for instance in StateFlow^{TM}/MATLAB/Simulink by MathWorks Inc.). All this engineer is going to create is essentially a reactive agent. Agent research has however shown that more sophisticated agents can be more adaptable and can have problem solving capabilities in real autonomous missions where advance specification of how to respond to all possible events of the environment is impossible or uneconomical to preprogram. Hence doing autonomous control via finite-state-machine definitions is viable but is a subset of more sophisticated agent based approaches.

One of the most sophisticated deliberative architectures are the belief-desire-intention (BDI) agents. These agents maintain a belief data base B that is regularly updated using sensors and signal processing and also process incoming communications events. A list of goals are maintained by the agent that contains the goals set by the human operator of the agent but the agent can extend the list by temporary goals that arise from its planning to achieve top level goals. Some events (that can be internal or external) and all goals are associated with actions sequences that are called plans to achieve them. In some common and fast executing BDI languages (such as AgentSpeak, Jason, Jack or Jade) all plans are to be declared by the agent programmer. Although this makes the agent less "creative" than logic inference based rational agents, as the plan set is fixed and not generated by the agent, they have the great advantage of fast execution and easier formal verification. In many safety critical systems such formal verification of a sophisticated BDI agent is crucial. Hence these kind of BDI agents combine the advantages of deliberative agents with the advantages of reliability that is fundamental for industry.

2 Control Planning Procedures

2.1 Preparing plan to go to a location

Sentences to use:

Prepare force sequence P to go to location Loc2 .

Available things are: Loc2(location) .

Details of the meaning:

This is a general procedure to obtain a sequence of pulse forces that moves the satellite from one location to another . The exact code goes as follows . Let P be a 'force sequence' . Recall control configuration C from memory . Let M be the 'force-to-thrusters conversion matrix' of C . Let B the 'thruster bounds' of C . Optimise fuel optimal force sequence P for thruster bounds B , matrix M to go from Loc to Loc2 using sampling period 3s . The action of 'preparing plan to go to a location' links up with jason statement 'cont.plan_approach' .

Resulting things are: P(force sequence) .

2.2 Preparing plan to go to centre

Sentences to use:

Prepare force sequence P to go to centre .

Details of the meaning:

This is a special case of getting a sequence of pulse forces to move the satellite from one location to another , namely the centre of the target box in this case . Let Loc0 be a 'location' . Set the 'vector' of Loc0 to "0,0,0" . Prepare force sequence P to go to location Loc2 . The action of 'preparing plan to go to centre' links up with jason statement 'cont.plan_approach_to_centre' .

Resulting things are: P(force sequence) .

Fig. 7. Part of an sEnglish document that declares usability of the sentences in a Jason agent's logic.

The following is part of the code of a BDI agent written in Jason that does deliberation in terms of having a set of goals, selects the most suitable intention for short term action and a plan that it executes before carrying on. It is about a GEO (geostationary orbit) satellite's control that keeps its position in term of latitude and altitude so that it can carry our communication functionality. The advantage of BDI agents in this context is that the agent can resolve complex problems of malfunction on board to serve its ultimate purpose of providing television and other broadcasting. The "// sE:" lines show sEnglish sentences that are explained in the sEnglish publication and are presented by a comment line of the corresponding Jason external call.

```
//==================================
// Jason/sEnglish SATELLITE CONTROL
// 25.4.2009, SMV
//==================================
// See the sEnglish document geo-pap.html
// PLANS
@homing !get_to_centre :
not_in_centre & at(Loc) <-
cont.plan_approach_to_centre(P,Loc) ;
!try_execute(P).
// cont.plan_approach -
// sE: Prepare force sequence P to go to location Loc2 .
// cont.plan_approach_to_centre -
// sE: Prepare force sequence P to go to centre .
(does not require naming the centre)
@exec !try_execute(P) : not_in_centre <-
?control_hw_changed;
!reconfigure_control_hw(C);
cont.apply_controls(P,C).
// cont.apply_controls(P,C)
// sE: Apply control control force sequence P
//using control configuration C with feedback regulation.
@control_hw !reconfigure_control_hw(C) :
control_hw_changed <-
cont.get_actuator_data(D);
cont.compute_control_configuration(C,D).
// cont.get_actuator_data(D) -
// sE: Get actuator data D from memory.
// cont.compute_control_configuration(C,D) -
// sE: Compute new control configuration C using actuator data D.
@emergency !send_sos : total_hardware_failure <-
cont.get_actuator_data(D);
com.send("MissCentre","all failed",D).
// com.send("M","all failed",D). -
// sE: Send to 'M' message text 'all failed' and actuator data D.
@totalfail +total_hardware_failure : true <-
get_thruster_rw_data(D);
cont.there_is_no_feasible_reconfiguration(D).
// comp.there_is_no_feasible_reconfiguration(D) :
// sE: There is no feasible reconfiguration for actuator data D.
//BELIEF UPDATES
@pos +at(Loc) : true <-
comp.distance(Loc) & D>2;
+not_in_centre.
// comp.distance
// sE: Compute the distance D of Loc from the centre.
```

The above Jason code illustrates in comments how the sEnglish paper with its sections links up with the logic of the agent code. Each "cont.action_name" or "com.messaging" external call in this Jason code has a corresponding subsection in the sEnglish document that unambiguously compiles into computer code. The agent reads the sEnglish document and its knowledge about control skills, world modelling skills and its behaviour constraints are updated in a way that it will have a well defined *shared understanding* with the operator-engineer who will also read the sEnglish document. This makes day to day practical work with the autonomous system not only safe but enjoyable for the engineers.

4. EXAMPLE: THE MISSION EXECUTION OF AN AGV

A webpage section that displays some AGV operations for "Preparing for mission" and "Executing the mission" is displayed in 8. The full sEnglish paper can be found a http://system-

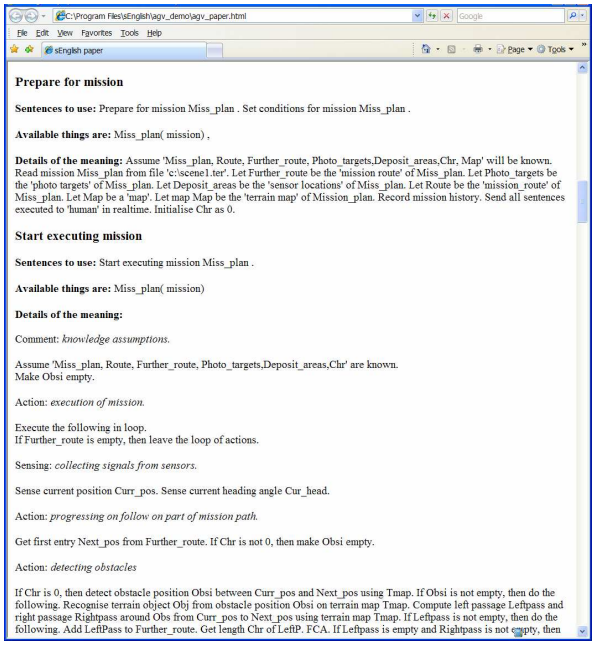

Fig. 8. Start of two sections on "Prepare for mission" and "Start executing mission" .

english.com [Sb]. The sentences used are concerned with mission data such as the route taken, plan of mission, list of photo targets, locations where sensors need to be deposited, etc. Also sensing of the environment and position is expressed in terms of sentences. Obstacle detection checks whether there is an obstacle on the straight line between the current position and the next way point. If yes then a path is worked out either to the left or right from the obstacle.

4.1 Description of the AGV movements problem

Fig. 10 displays the laboratory environment where the AGV needs to move around. The model houses are near to each other. At some places a passage is only twice or three times the width of the AGV. On the other hand the AGV is not able to turn on a smaller radius than 3 times its width. Hence its steering

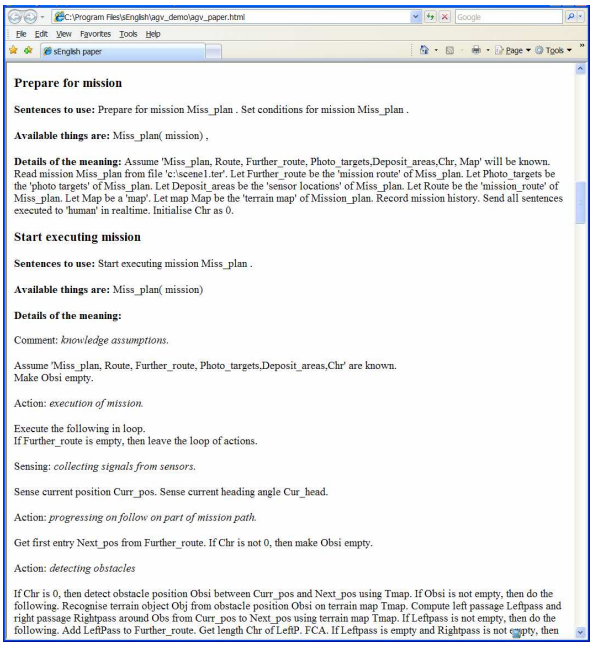

Fig. 9. Some of the mission description document in sEnglish.

is seriously limited. As a mission continues, the battery of the AGV provides less and less power that affects the speed and angle of turning of the vehicle while the same control signal is used from the computer. This of course necessitates the use of feedback that in this example is put into vehicle operations in sentences. The AGV uses camera based navigation system.

Fig. 10. The environment where the AGV (lid removed in this picture) needs to navigate and pass through among houses.

The AGV has four onboard cameras with servo controls of their viewing directions. In this demo also an external overhead camera was used to detect the location of the AGV within the environmental model and relayed back to the vehicle via a wireless network communication. Heading angle of the vehicle is estimated from past sensed positions and steering angle records. Inertial measurement units (IMUs) can be easily integrated to enhance te precision and reliability of vehicles navigation All data fusion and world modelling is ”programmed” in sentences.

4.2 Path planning and execution skills

There is a considerable amount of literature available on path tracking of autonomous or semi-autonomous vehicles. Although [Willms] deals with environmental constraints, direct application of this scheme was not possible as the future path of the vehicle is seriously limited by its current heading. What matters is that passing through gaps can only be done by starting from a bounded locality. This necessitates more careful planning and an arbitrary way point sequence among the houses is not executable at all. The sEnglish paper of the AGV in Fig. 10 at http://system-english.com contains sections that describes the path execution of the AGV in English sentences that is easy to understand and debug. Causes of possible problems in the course of a mission also become clear to the operators, much more so than that is possible with ordinary programming that separates meaning from what the program does.

5. COMPARISON OF SENGLISH WITH OTHER CONTROLLED ENGLISH TEXTS

This section reviews the best and most known controlled languages.

5.1 Comparison with Attempto

Attempto (ACE) [Attem] is a controlled English that compiles restricted English text into first order logic (FOL), PQL, FLUX, RuleML or to web ontology language OWL descriptive logic (DL) that are formal knowledge representation languages. The semantics of these formal languages need to be further defined before sentences can have a meaning for a machine. In fact this translation exercise into a formal language is only scratching the surface of what one may call agent's understanding. World modeling, planning of actions and decision making is still a major remaining task.

As opposed to Attempto, sEnglish sentences compile into a high level language to form a ”meaning”. If sEnglish is used for communication then functions can be defined in this high level language to carry out world model modifications by the agent. This is a more direct approach to what an agent needs.

Yet the most important difference between Attempto and sEnglish is that

(1) Attempto is a language intended for formal world model descriptions and communications, it would be difficult and inefficent to write computer programs and procedures in Attempto.

(2) sEnglish is primarily intended for describing procedures, system operations, it is primarily for programming.

These fundamental differences can make Attempto a complementary system sEnglish of knowledge representation to build a joint powerful systems where underlying functionality is defined in sEnglish. Such a system can describe procedures and communication procedures of engineering systems more effectively, especially for autonomous systems or distributed engineering projects that involve large teams of human developers.

Syntactic differences are:

(i) Attempto uses basic grammar for sentence interpretation rules, sEnglish uses very small amount of grammar and recognizes sentences by their definition forms.

(ii) Attempto uses databases for word classes such as verbs, nouns, adverbs, pronouns, etc. to interpret sentences. sEnglish only needs a set of sentences each defined by a set of other sentences and it also uses an ontology of concepts that can consist of several words.

5.2 Comparison with Common Logic Controlled English (CLCE)

John Sowa's CLCE [Sowa] is a way to state logic formulas in English. Under certain conditions the translation is bidirectional, i.e. from first order logic (FOL) to English and from restricted English to FOL. This makes CLCE especially suitable to articulate mathematical relationships precisely in English. As ACE also compiles into FOL, CLCE is aimed at describing modelling relationships in a formal manner. Therefore the comparison points made above between Attempto and sEnglish remain valid:

(1) CLCE is a language intended for formal world model descriptions and communications, it would be difficult and inefficient to write computer programs and procedures in CLCE. On the other hand sEnglish is primarily intended for describing procedures, system operations, it is primarily for programming.

(2) CLCE uses basic grammar for sentence interpretation rules, sEnglish uses very small amount of grammar and recognizes sentences by their meanings.

(3) CLCE can usefully rely on databases for word classes such as verbs for predicates, nouns for variables and prepositions for special predicates to interpret sentences. sEnglish operates by a self contained list of sentences each defined by a set of other sentences and it also uses an ontology for concept names that can consist of several words each.

5.3 Comparison with Processable English

Similarly to Attempto and CLCE, Processable English [Peng] translates a syntactically correct English text into first order logic (FOL) statements. Hence some of the comparison statements made above for Attempto and CLCE, are also valid for the comparison of PENG and sEnglish.

To summarize, the main differences are:

(1) sEnglish has relatively small amount of grammar relative to ACE, CLCE or PENG.

(2) sEnglish focuses on meaning definition by other sentences when defining a sentences, not on forming a sentence that permits a logic representation as in ACE, CLCE and PENG. Each sEnglish sentence has a corresponding high level program code.

(3) sEnglish relies on meaning of sentences in terms of other sentences. In any text each sentence ultimately compiles into a high level code that is a function with input-output objects.

6. CONCLUSIONS

This paper describes a "publishing" system for autonomous systems where the documents are written in English and the vehicles can read them to improve their navigation, sensing and control skills with regards to self-movement and manipulation of external objects. They can also read behaviour rules and limitations. The advantage of using a publishing system over traditional "reprogramming" are

- The human users will share the knowledge of the agents and that reduces misunderstanding when complex intelligent behaviour is needed during autonomous missions.
- The "published" papers can be distributed to agents of a specific control program that define their deliberative or reactive behaviour. Instead of reprogramming, the autonomous systems learn from publications as humans do.
- The sEnglish publications can be distributed within a company or on the Internet as "proper publications" to make colleagues aware of results in sensory signal processing, navigation, environment modelling and adaptive/learning control methods.

This paper illustrated the system on a small autonomous ground vehicle. The system can also be used with AUVs, autonomous UAVs, spacecraft and in most autonomous robots. Further background reading is [Veres].

7. REFERENCES

[Attem] K Kaljurand, N E Fuchs (2007). *Verbalizing OWL in Attempto Controlled English*, Verbalizing OWL: Experiences and Directions (OWLED 2007).

[Molnar] Levente Molnar and S M Veres (2009). System verification of autonomous underwater vehicles by model checking. *Oceans'09, 11-14 May, Bremen, Germany.*

[Peng] Rolf Schwitter (2008). Processable English, http://wikipedia.org *Department of Computing, Macquarie University, NSW 2109, Australia*

[Sowa] Sowa, John F (2008). *Common Logic Controlled English (CLCE)*, http://wikipedia.org .

[Sb] (2009). *sEnglish Reader Agent and Authoring Tools*. SysBrain Ltd. London, UK, 3 More London Riverside, SE1 2RE, http://system-english.com.

[Veres] Sandor M Veres (2008).*Natural Language Programming of Agents and Robotic Devices*. SysBrain, London, ISBN 978-0-95584417-0-5.

[Willms] Allan R Willms and X Yang Simon (2008). Real-time robot path planning via a distance-propagating dynamic system with obstacle clearance. *IEEE Transactions on Systems, Man and Cybernetics* **38**(3),884-893.