The 9th International Symposium on Automation and Robotics in Construction June 3-5, 1992 Tokyo, Japan

Controlling an Intelligent Excavator for Autonomous Digging in Difficult Ground

Derek Seward, David Bradley, Jim Mann and Mark Goodwin
Department of Engineering
Lancaster University
Lancaster LA1 4YR
United Kingdom

ABSTRACT

This paper reports on the recent advances made in developing an autonomous robot excavator. Previous work on a fifth-scale model was reported at earlier symposia, but the technology has now been transferred to a real excavator - LUCIE - **L**ancaster **U**niversity **C**omputerised **I**ntelligent **E**xcavator. The paper concentrates on the architecture of the software control which enables the machine to modify its behaviour to cope with highly varying ground conditions. A single powerful processor controls the low-level motion of the excavator arm, as well as high level tactical and strategic behaviour. An A.I. rule-based approach has been adopted for high-level functions. Successful field trials are reported.

1. INTRODUCTION

Early attempts to automate the excavation process were only partially successful because of the lack of intelligence within the control system, which is necessary to cope with variations in ground conditions. It is clear that the conventional robotic approach, of moving through a pre-determined path in space, is not subtle enough for efficient digging. To address this problem, two steps were taken. Firstly, a grant was obtained from the Science and Engineering Research Council for site observations of the techniques used by expert excavator drivers [1]. Secondly, a fifth-scale working model of a back-hoe excavator arm was constructed in order to provide a laboratory facility for investigating digging tactics and strategies [2],[3]. *Figure 1* shows the model in use.

Figure 1
The fifth-scale model in action

More recently, support from JCB Research Limited has enabled the technology to be transferred to a full-size tracked mini-excavator - LUCIE - see *figure 2*.

In order to give the project focus, the following aims were adopted:-

- The project will concentrate on the intelligent control of the excavator arm only, and **not** consider the movement of the vehicle itself around a building site.
- The selected task is **trenching**. The aim being to dig a flat-bottomed trench accurately to a depth input by the user.
- The machine must dig efficiently in varying ground conditions, without human interference. This includes coping with obstructions such as boulders.
- The machine must be self-contained and not require an external computer to be connected.

The fundamental measure of success is the degree to which:-

"The bucket is filled in the shortest time possible"

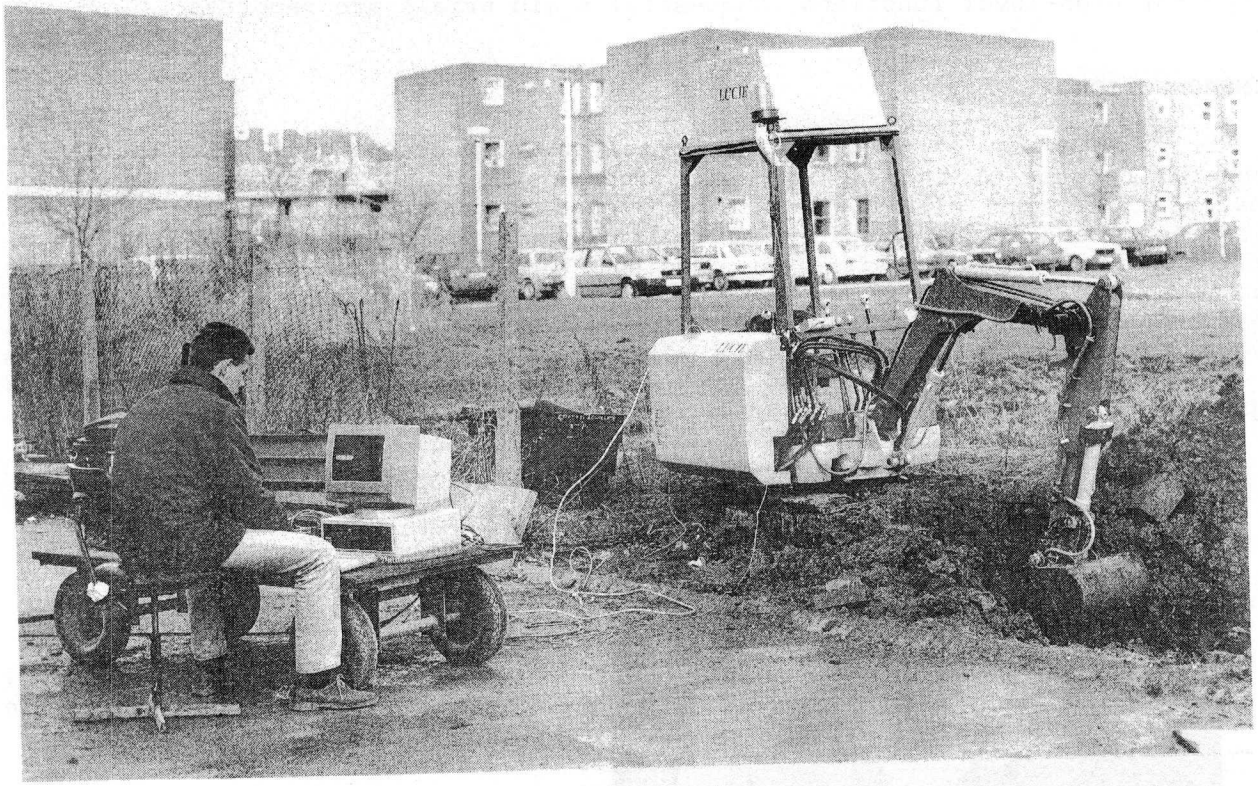

Figure 2
LUCIE in "teach" mode

2. HARDWARE

It is not the purpose of this paper to describe the system hardware in any detail, and this has been done elsewhere [4],[5]. The aim has been to keep the system on the full-sized excavator as close as possible to that used on the fifth-scale model. Briefly, as with any mechatronic system, there are three major components:-

- A sensing system, which on the full-sized excavator consists of a rotary optical encoder on each joint and a tilt sensor on the vehicle body. A simple user interface instructs LUCIE on the depth of trench to dig.

- An effector system, which consists of hydraulic gear pumps supplying proportional electro-hydraulic valves, which drive conventional double-acting cylinders.

- A control system, which is based around a Harris RTX2000 processor. This is a fast and powerful processor that is designed to be programmed in the language FORTH.

The philosophy adopted in building the hardware was to speed up development time by buying off-the-shelf components wherever possible. This has resulted in an effective system for research purposes but it bears little resemblance a production model.

3. SOFTWARE

The key to successfully competing with a human operator lies in being able to adapt to the prevailing ground conditions, which are of course usually unknown at the start of the task. An early decision was to divide the software into two parts with a clean break between them. The first part is a low-level controller which drives the excavator bucket tip in the desired direction. The second part is a high-level rule-based controller which provides the intelligence. Each of these parts will now be considered separately.

3.1 The low-level controller

The purpose of this part of the software is to control the proportional elctro-hydraulic valves in such a way that the excavator bucket moves in the manner demanded by the high-level controller. In order to achieve the necessary degree of accuracy ($\pm$ 25 mm) it was judged that **closed-loop** control was required, with feed-back from the joint encoders being used to correct the trajectory. On LUCIE, the digital encoders operate on an interrupt basis to the main Harris processor. The low-level controller must:-

(1) determine which joint has moved
(2) decide on the direction of movement
(3) count the number of signals
(4) convert this to an angle change

A major decision was made concerning the basic mode of operation of the controller. There are two principle approaches. Firstly, most conventional robot controllers operate on some form of **point-to point** control. The end effector is instructed to move to a particular set of spatial co-ordinates - e.g. GOTO(x,y,z). This is normally done by using inverse kinematics [2] to calculate the required joint angles. If, as in this case, the path taken is important, as well as the final destination, then the calculations need to be repeated for a large number of intermediate steps.

Secondly there is **velocity** control. This is more of a "fly-by wire" approach, in which the task of the controller is simply to achieve motion in a particular direction at a particular velocity - e.g. GO(q,v). From a control point of view, this is best achieved using a **target point**. An imaginary point is moved with the desired velocity vector, and the error between it and the actual position of the bucket tip provides the feedback for closed loop control. The trigonometric calculations involved are much simpler as it only involves forward kinematics, and this is clearly an advantage in a real-time application..

The decision was made to adopt the second approach. The main justification being that, when moving in the ground, the variable nature of the soil resistance makes it unlikely that a specified point can be reached - there is therefore little point in performing the necessary calculations. Observation of human

operators indicates that this is the time when a high degree of adaptability is required in order to gain efficient operation. It is also closer to the form of control used by human operators and permits the rules in the high-level controller to be written in a much more humanistic form. For movement in air, which is much more predictable, it has been found that a point-to point controller has significant advantages in terms of speed and precision. It is an easy matter, however, to convert the velocity controller to provide pseudo point-to point behaviour.

Separate control loops were provided for the arm joints, the bucket angle and the slew angle. A three term PID controller was implemented, and the basic structure is shown in *figure 3* . It was found that adequate performance can be obtained by running the controller at 40 Hz.

A significant bonus that is gained from the target point approach is that the size of error detected is proportional to ground resistance. It therefore provides a form of force feedback, and this vital piece of information is communicated to the high-level controller where it becomes the key for determining intelligent behaviour.

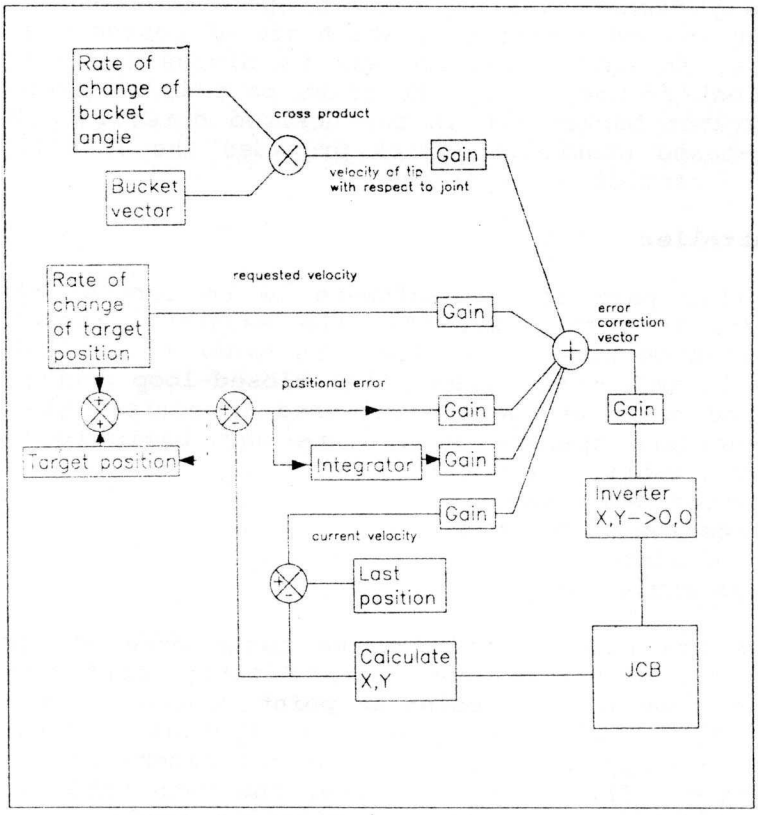

Figure 3
Block diagram of control strategy

3.2 The high-level controller

The high-level controller is where the intelligence resides. The **task** of digging a trench was decomposed into a series of **activities**. These are shown in ***figure 4***. The high-level controller can therefore be thought of as an **activities manager** which defines and orders the individual activities that go to make up a task. As stated above, the key to the intelligent adaptive behaviour is the use of positional error from the low-level controller. This is used as the **trigger** to switch between activities when operating in the ground. For example, if the activity "penetrate" is running, the bucket teeth are driven into ground and the positional error builds up as the ground resistance increases. When the error reaches a specified value the switch to "drag" is triggered. The term "drag" is used to describe the activity of scraping the bucket horizontally towards the cab of the excavator. (If the "penetrate" mode was continued for too long then it could cause the whole machine to tip over). Likewise, during drag, if an obstruction, such as a boulder, is encountered the error builds up and this triggers an appropriate behaviour change.

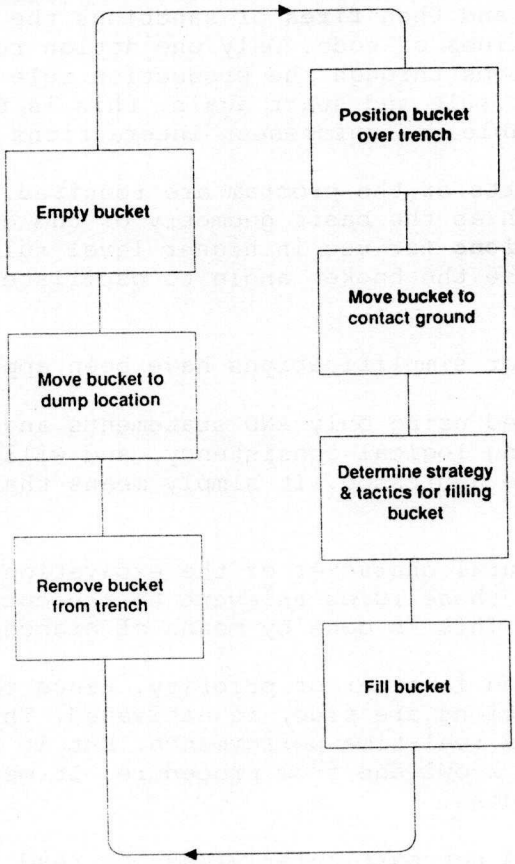

Figure 4
Activities that make up the trenching task

With a subtle procedure, such as excavation, it is not possible to simply write the software instructions at the terminal and expect them to work efficiently at the first attempt. There is an inevitable learning and optimisation process to be carried out, and this can only be done by actual physical trials. (Computer simulation would seem to be of little value in this case, however learning through **neural network** techniques looks promising for the future.) It follows that the software environment should be one in which it is relatively easy to understand and change the behaviour of the machine. This points towards a rule-based approach, and that adopted is the well known AI technique - a **production system**.

Briefly a production system is divided into three parts:-

- The **production rule memory** which contains all the rules which determine the behaviour in a form such as:

  IF (condition is true) THEN (do action)

  A typical rule in our case is:

  IF (penetration of bucket > 300 mm) THEN (rotate bucket and go to tip)

- The **working memory** which contains all the current values of the variables used for checking the conditional part of the rules. The value of these variables can be changed by the low-level controller, certain rules in the activities manager or external sensors.

- The **inference engine** which loops through the rules, checks the conditions, decides priorities and then **fires** or sanctions the action. In practice this may be just a few lines of code. Only one action rule must be executed each time the program loops through the production rule memory. It must then return to the first rule and start again. this is to ensure determinacy, and to avoid unpredictable and unforeseen interactions between rules.

In our case two other parts of the program are required. Firstly a section containing constants such as the basic geometry of the excavator, and secondly a library of low level **actions** for use in higher level rules. A typical action is "wiggle", which will cause the bucket angle to oscillate as an aid to penetration.

Three other refinements or simplifications have been applied:-

1. Rules can be compounded using only AND statements and **not** OR statements. This is for reasons of ensuring logical consistency, and will be important when issues of safety criticality are addressed. It simply means that extra rules are required.

2. Because of the procedural character of the excavation process, the rules are partitioned so that only those rules relevant to the current activity are visible to the inference engine. This is done by means of branching rules.

3. The rules are tabulated in order of priority. Hence the first rule encountered, whose conditions are true, is activated. This clearly simplifies the inference engine and aids real-time performance, but it is opposed to the AI philosophy of separating knowledge from procedure. It means that greater care is needed when adding new rules.

The current FORTH code is not particularly easy to read, partly because it uses Reverse Polish Logic. Work is currently underway to produce a programming environment were rules can be written in a more natural form before compilation into the Reverse Polish form. *Figure 5* shows a small section of the activities manager production system together with a concurrent section of the working memory. The safety section will grow considerably as more sensors are added to monitor the working environment.

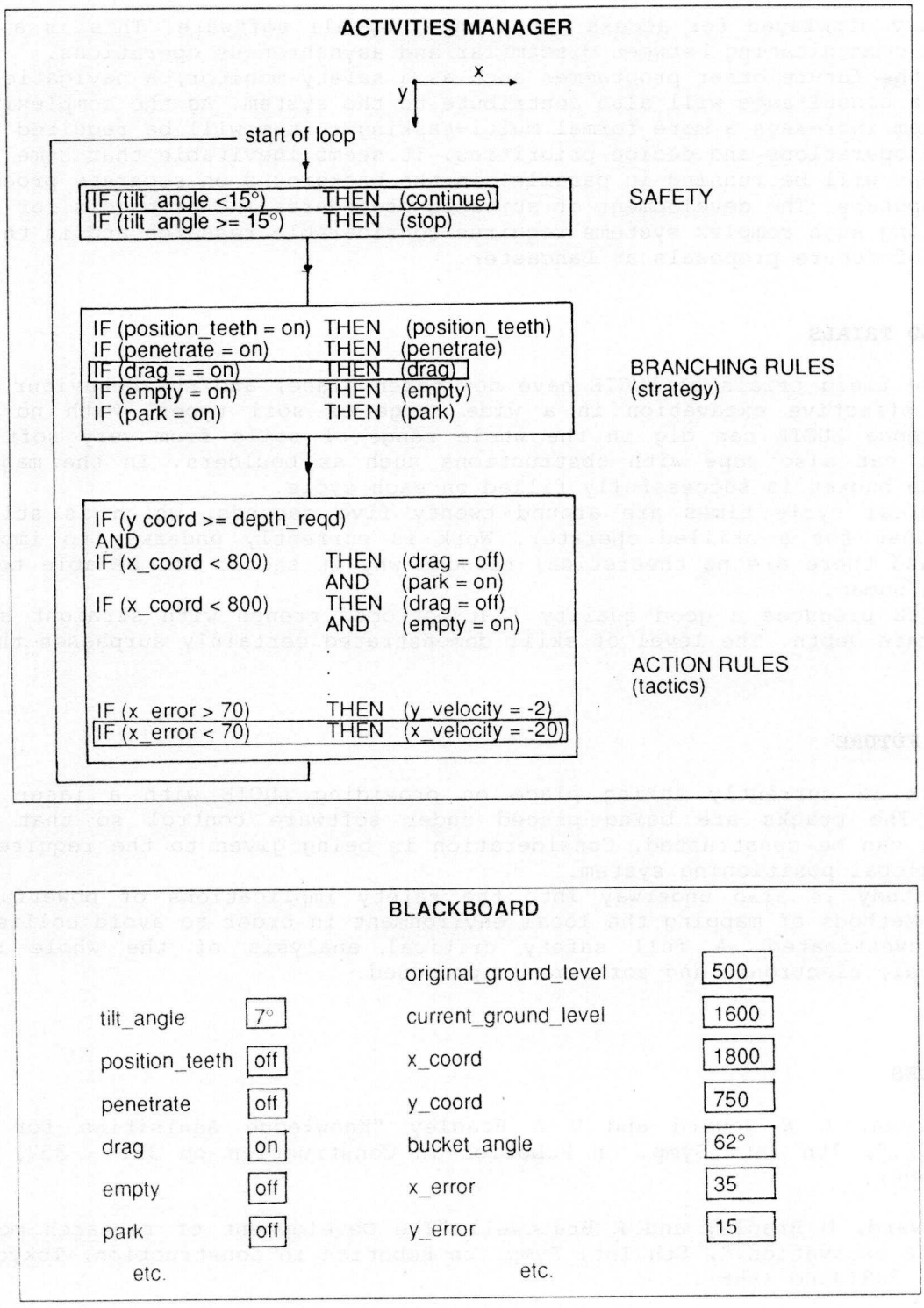

Figure 5
Software structure for production system and current contents of working memory "blackboard"

3.3 Communication

Communication between the high and low-level controllers can be thought of conceptually as a **blackboard**. In other words the contents of the working memory

are openly displayed for access (or change) by all software. This is a flexible means of communicating between dissimilar and asynchronous operations.

In the future other programmes such as a safety monitor, a navigation module or expert consultants will also contribute to the system. As the complexity of the system increases a more formal multi-tasking system will be required to co-ordinate operations and decide priorities. It seems inevitable that some operations will be running in parallel in the background on separate processors or transputers. The development of suitable procedures and protocols for integrating such complex systems requires considerable research and is the subject of future proposals at Lancaster.

4.0 FIELD TRIALS

Extensive field trials of LUCIE have now taken place, and the behaviour tuned to produce effective excavation in a wide range of soil types. With no operator interference LUCIE can dig in the whole range of soils from very soft to very hard. It can also cope with obstructions such as boulders. In the majority of cases the bucket is successfully filled on each cycle.

Typical cycle times are around twenty five seconds, which is still about double that for a skilled operator. Work is currently underway to improve the speed, and there are no theoretical reasons why it should not be able to compete with the human.

LUCIE produces a good quality flat-bottomed trench with straight sides and of accurate depth. The level of skill demonstrated certainly surpasses the novice driver.

5.0 THE FUTURE

Work is currently taking place on providing LUCIE with a laser guidance system. The tracks are being placed under software control so that complete trenches can be constructed. Consideration is being given to the requirements of a full global positioning system.

A study is also underway into the safety implications of powerful mobile robots. Methods of mapping the local environment in order to avoid collisions are being investigated. A full safety critical analysis of the whole system - mechanical, electronic and software is proposed.

REFERENCES

1. P Green, D W Seward and D A Bradley "Knowledge Aquisition for a Robot Excavator.", 7th Int. Symp. on Robotics in Construction pp 351 - 357, Bristol, (June 1990).

2. D Seward, D Bradley and R Bracewell "The Development of research models for automatic excavation.", 5th Int. Symp. on Robotics in Construction, Tokyo, Vol 2, pp 703 - 708(June 1988).

3. "Control and operational strategies for Automatic Excavation", with Bradley et al, 6th Int. Symp. on Robotics in Construction, San Fransisco, June 1989.

4. R H Bracewell, D A Bradley, R V Chaplin and D W Seward, "Control Systems Design for Robotic Backhoe.", 7th Int. Symp. on Robotics in Construction pp 222 - 229, Bristol, (June 1990).

5. D W Seward, "LUCIE - The autonomous excavator". Industrial Robot International Quaterly Vol 19 No 1 pp 14 - 18, (March 1992).