Computer-Aided Production Engineering, edited by V.C. Venkatesh and J.A. McGeough
Elsevier 1991

Towards the flexible manufacture of food

K. Venuvinod Patri
Department of Manufacturing Engineering, City Polytechnic of Hong Kong

Abstract

The nature of an ideal programmable kithen system (PKS) for cooking fresh food at homes is discussed. The development of an experimental Home PKS built around a Puma 760 robot using flexible manufacturing cell strategies is described. The importance of the concepts of basic units for materials and the use of safe points and canned subroutines for the robot is demonstrated. Finally, the syntax of an approach for coding recipes into PKS-hardware-mdependent control programs is presented.

1. INTRODUCTION

Clearly, of all manufacturing processes, cooking food is the most ancient and extensive. Every year over a trillion fresh meals are cooked in households and restaurants around the world. Thus, cost-effective automation of cooking has the potential of releasing an enormous number of man-hours and, more importantly, woman-hours each year from the chore of cooking. However, notwithanding its immense economic and social significance, progress towards the automation of cooking fresh food has been extremely slow. There are two main reasons for this.

Firstly, fresh food is almost invariably cooked in large variety and small batches (the mass production of processed/canned food is excluded in this discussion). In home kitchens the batch size is often down to one so that there is a need to adopt flexible (i.e. programmable) automation techniques. However, unlike hard automation, flexible automation techniques are only just maturing even in industrial production where the potential for heavy capital injection is higher.

Secondly, a quantitative understanding of the science of cooking is conspicuous by its absence. Even today, we are unable to specify or monitor in quantitative terms the basic attributes of cooked food such as taste, flavour and texture. owing to the relative absence of 'transducers' for sensing these attributes, one still has to resort to the use of human sensory organs. It is not surprising therefore that cooking continues to be relegated to the realm of art.

Consider now the nature of flexible automation techniques needed in home kitchens Cooking essentially involves the transportation and mixing (assembly) of measured quantities of ingredients (raw materials) followed by their mechanical and thermal processing The processes are controlled by monitoring the product attributes such as taste flavour and texture. Automation of cooking therefore requires the automation of the processing, transporting, dispensing, assembly and monitoring operations involved.

Concerning processing operations, a wide range of sophisticated processing equipment such as juicers, microwave ovens, hot plates, etc. are already being used extensivey in affluent households. These could be easily interfaced to a personal computer so tha the various processes could be under programmable control.

An analysis of the common range of operations in cooking shows that the majority of transportation and processing (e.g. shallow-frying) operations could be effectively performed by robots operating in the pick and place and the continuous path modes respectively. In contrast, from an economic point of view, the use of robots is a problem since the cost of robots is still too high to be within the reach of households. However, once the technological feasibility of Programmable Kitchen Systems (PKS) is firmly established, robot costs are likely to fall significantly since, spurred by the huge demand from household consumers, robots could be mass produced.

It is on the monitoring and inspection front that the greatest hurdles to PKS lie. Given the absence of commercial transducers to sense product attributes, one has to largely resort to 'open loop cooking'. However, with careful standardisation of cooking ingredients and process parameters it should be possible, in principle, to produce a fairly large variety of food of acceptable quality through 'open loop cooking'. Further, the PKS software can be so designed as to let the user act as the feedback transducer. He cantest the food in the first round and adjust a given set of input parameters in software to suit his needs. For example, he could set the oven-on-time shorter to make his steak rarer.

Several workers in the field of robotics and artificial intelligence have acknowledged cooking as a very challenging task for robots. For instance, Ayres [1] has noted that a number of kitchen operations (such as washing dishes individually, cutting meat, cleaning fish and shrimps, and, separating crab meat from shells), although simple for humans to perform, are extremely difficult for robots. Likewise, Agre [2] refers to several kitchen operations while illustrating his concepts on how robots could be equipped with comon sense. Asimov and Frenkel refer to Engleberger's opinion that "one day industrial and personal robotics will converge" and that, in the not-too-distant future, "homes will have a great, big massive computer in the robot pantry" [3]. Safford [4] describes a few conceptual layouts of automated kitchens involving voice input, conveyors and multi-armed robots. More recently, a robot for pizza making, called the Pizzabot, has been developed by the Carnegie Mellon Research Institute's Centre for Human Services Robotics (CHSR) [5]. This paper describes the work carried out by the author and his students at the Hong Kong Polytechnic, which, it is believed, is the first major effort in examining the technical feasibility of the concept of Home PKS.

2. THE EXPERIMENTAL PKS AND CONTROL STRATEGIES

Figure 1 shows the nature of an 'ideal' home PKS. An interactive menu driven program enables the user to select dishes, the number of servings, and the time when the food is required to be served. Once the user has input the order, the computer calculates the types and quantities of the ingredients required and prompts the user to top up the ingredients to the minimum levels at appropriate locations on the PKS layout displayed on the graphic screen. The computer calculates the time required for each of

the cooking operations, determines the operation schedules and starts cooking at the point in time that ensures that the food is ready and served at the exact moment requested by the user. Cooking hygiene is ensured by using appropriate materials for the parts of the robot gripper that come in direct contact with the cooking ingredients, and programming the robot to clean its grippers at a rinsing station between operations. Human safety may be ensured by moving the robot to a parking station and inactivating it whenever someone enters the kithcen.

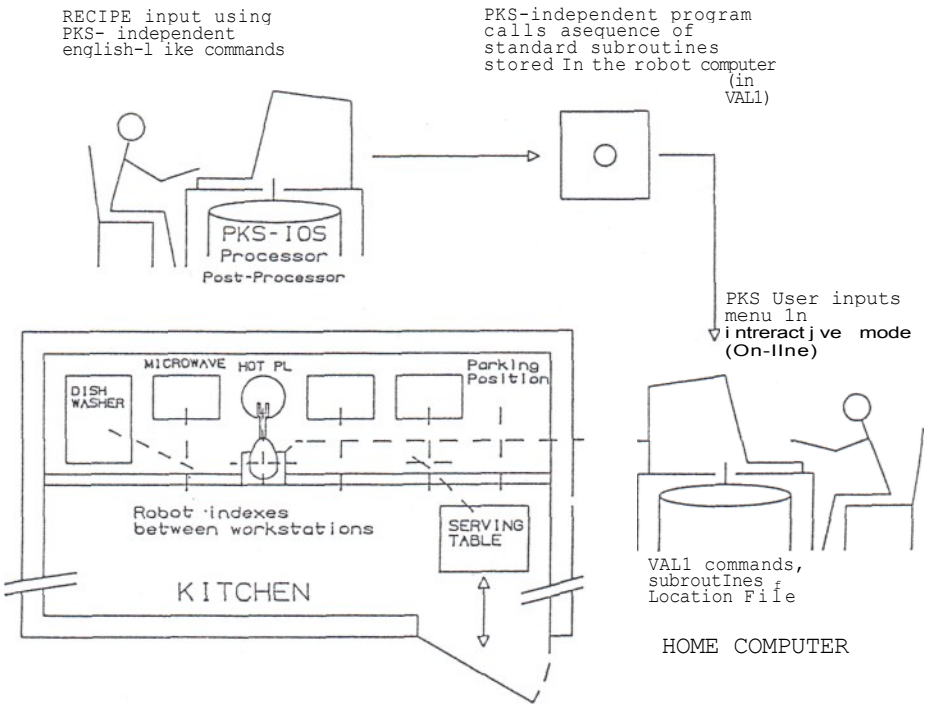
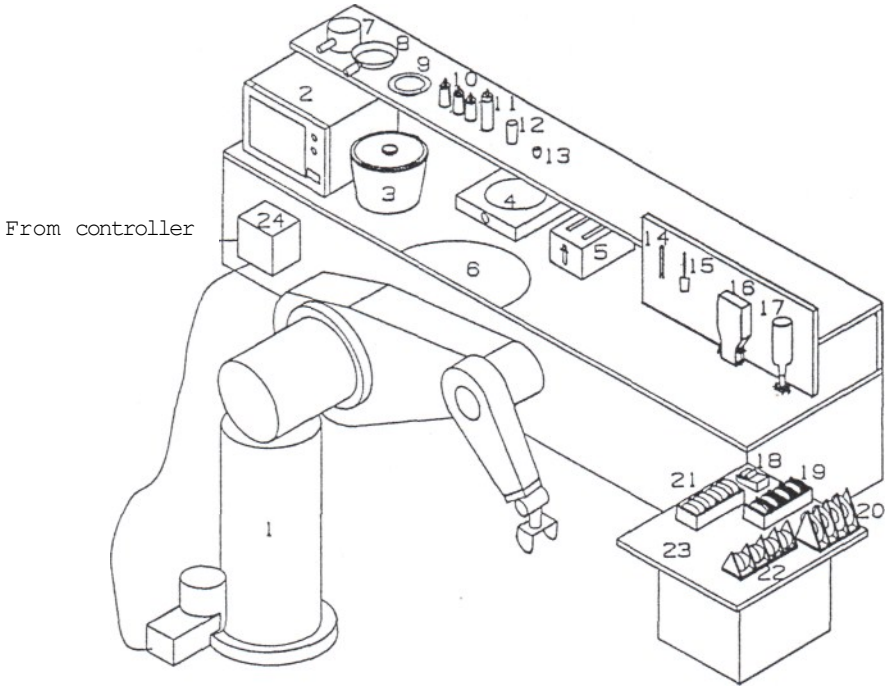


Figure 1. The Concept of the Ideal PKS

The lay out of the experimental PKS is shown in Figure 2. A primary requirement of any PKS is that all processing equipment must be controllable from a central computer. In the case of the Experimental PKS, the PUMA Computer acted as the central computer. A specially built relay box was used to interface the cooking equipment with the output ports of the PUMA robot controller. Each functional switch or contact of the cooking equipment was controlled by a separate relay contact. Timing functions were achieved by controlling the duration for which a signal was held.



1. PUMA 2. Microwave 3. Rice Cooker 4. Hot-plate
5. Toaster 6. Work Area 7. Pot 8. Pan
9. Bowl 10. Salt/Sugar/Seasoning 11. Oil
12. Glass 13. Cup 14. Tongs 15. Turner
16. Rice Hopper 17. Water 18. Sausages 19. Noodles
20. Plates 21. Pork 22. Hamburgers 23. Service Area
24. Relay Box

Figure 2. The Experimental PKS

Consider now the classification of the cooking ingredients (materials) used in the Experimental PKS. At the highest level, these materials are classified into 'Principal' or 'Auxiliary' materials. Principal materials are those which establish the uniqueness of a dish. Auxiliary materials are those which are used in a range of dishes and do not establish the uniqueness of the dish. For example, a pork chop is identified as such

because of the pork rather than the salt which goes into it. Thus, pork belongs to the principal class whereas salt belongs to the auxiliary class.

The distinction between principal and auxiliary materials is more than semantic. Auxiliary materials are repeatedly used in a range of dishes. Hence they can be located at fixed permanent positions (on the upper rack in the Experimental PKS). Further, these materials can be dispensed in quantities that are integral multiples of standardised 'Basic Units' (BU) discussed later in this section. In contrast, principal materials vary from dish to dish and, hence, cannot be assigned permanent locations. Moreover, the determination of the BUs for these is an involved process requiring significant future research.

Consequently, in the Experimental PKS, solid principal materials (e.g. sausage, hamburger) were pre-processed to the required size and presented to the PKS. For granular principal materials (e.g. rice) and auxiliary materials, a better approach was considered to be the establishment of a 'Basic Unit' (BU) for each material such that the requirement of the material in any conceivable dish could be expressed as a multiple of the corresponding BU. The BUs for the materials in the Experimental PKS were determined through a careful investigation of the practices adopted by 'mums at home' in cooking various dishes. Thus, for instance, cooking *oil,* which is a *liquid* type *auxiliary* material, had a BU *of 5.00 ml* and was stored at location *B3.* The key data (shown in *italics)* were stored in the appropriate material and location files.

An advantage of the control of materials in terms of their BUs is that this enables the use of simple and commonly available 'digital' dispensing devices and facilitates robotic operation. Thus, for instance, the sugar dispenser used was a commonly available bottle that could dispense a fixed quantity of powder each time it was turned upside down whereas the liquid dispenser was an adaptation of the 'liquour dispenser' used in most bars. Such 'digital' devices are not only inexpensive but enable easy programming of the robot for dispensing one BU. When multiple BUs were required, the robot merely repeated the dispensing cycle the required number of times.

The tools in the PKS were classified as *1. Material Storage Tools* for principal materials (e.g. racks for holding sausages) and auxiliary materials which could be fixed in location (e.g. the rice hopper) or movable (e.g. the salt bottle); *2. Mechanical Processing Tools* (e.g. tongs or spatulas for frying, stirring, etc.); *3. Ingredient Transporters* (e.g. a cup for transporting water); *4. Work Pallets* (e.g. pans for holding material during assembly and processing); and 5. *Service Pallets* (e.g. plates for presenting the cooked food on the service table). Information on the types and permanent storage locations for tools were stored in a Tool File according to this classification. It is seen that the classification is intimately linked with the way the tools interact with cooking materials during the operation of the PKS. These tool/material interactions, in turn, largely determine the classification of robot operations discussed later. Since the ideal Home PKS must be suitable for both human and robotic operation, an effort was made to select the tools that are commonly found in home kitchens and then make minor modifications to them to facilitate robotic handling.

Consider next the control of the 'process operations', i.e. the key 'cooking' operations, in the PKS. Examples of such operations are boiling, roasting, baking, shallow-frying and deep-frying which determine the uniqueness of a dish. Almost invariably, these operations involve heating the material(s), (thus, assembly of ingredients, as in a tossed salad, is excluded from this class of operations).

The major factors affecting the control of process operations are the required equipment settings (e.g. intensity of hot-plate) and the processing time (e.g. duration for which a sausage is to be shallow-fried). These control parameters vary from dish to dish and need to be determined through scientific experimentation and stored in the database of the PKS. In the case of the Experimental PKS, a range of dishes covering a variety of process operations were selected and the processing parameters associiated with each operation in each dish, as adopted by 'mums at home', were determined through stop watch studies. In some cases, the cooking procedures recommended in recipe books were adopted after fine tuning and validating the process parameters through experimentation.

As described earlier, the control of process operations was achieved by hard-wiring the processing equipment to the I/O ports of the PUMA controller through a purpose built relay box. The operations could then be controlled in software through appropriate 'SIGNAL +. n' commands in VAL1.

Table 1 shows the classification of robot operations in the Experimental PKS. Here, operations belonging to the 'processing' type refer to those required in the manipulation of processing equipment (e.g. close the lid of the rice-cooker). The explanations for the 'dispensing', 'transfer material' and 'transportation' operations are included in Table l.

3. PKS-SPECIFIC RECIPE PROGRAMS

The execution of each specific recipe requires a distinct combination of process and robot operations. While some of these robot operations could be specific to the recipe,it turns out that the majority appear in the execution of a range of recipes. An example of such a common operation is 'get oil bottle' which could appear in 'fried sausage' as well as in 'boiled noodle'. This feature was utilised in the Experimental PKS to create an efficient programming environment.

An aim of the Experimental PKS was to create a programming environment which enabled the quick creation of a new recipe program by calling the appropriate sequence of sub-routines stored in the memory of the robot controller. This was achieved as follows:

(i) The location coordinates of the storage point of each tool and processing equipment were stored in a Location File,

(ii) In addition, a set of 'Global Safe Points (GSFPs)' within the global work space of the robot was specified such that robot movement between any arbitrary pair of GSFPs was executable by a simple MOVE command in VAL1 without the danger of collision,

(iii) Likewise, a set of 'Local Safe Points (LSFPs)' within the local working space of each robot operation was specified such that a simple MOVES command in VAL1 between any relevant pair of LSFPs could be executed without the danger of collision,

(iv) The location coordinates of all GSFPs and LSFPs were stored in the Location File.

(v) Next, a sub-routine was written in VAL1 for each robot operation such that all robot movements within the operation, except the first and the last movement, were executed between LSFPs; either the operation began and terminated at the same GSFP, or, wherever this was not possible, precedence relationships between operations were specified such that the associated chain of operations began and terminated at the same GSFP; and the robot speed for the operation was specified within the sub-routine.

Table 1. Classification of Robot Operations

Processing : Thermal : e.g. OPOV { open oven door }
Processing : Mechanical : e.g. FRY { fry in pan }
Dispensing : from movable dispenser : e.g. OIL { add oil, duration 35 sec }
Dispensing : from fixed dispenser : e.g. HOPPER { dispense rice, 1 BU }
Transfer Material : e.g. POPO { pour from cup }
Transportation [arguments : of from to] : e.g TONG1 [noodle STP pot
---

Notes : STP = Storage Point, NSFP = Nearest Safe Point

Once the sub-routines were written, the task of developing a recipe program was reduced to calling the appropriate sequence of sub-routines. As an example, the recipe program called NOODLE, written in VAL1, is reproduced below : " 1. *remark boil noodle 2. speed 20.00 always 3. moves SAFE1 {go to location SAFE1} 4. signal 5,,,,,, {switch on hot plate} 5. gosub POT {place pot on hot plate} 6. gosub CUP A {get cup} 7. gosub WATER {dispense water} 8. gosub POPO {pour water from cup into pot} 9. gosub WATER 10. gosub POPO 11. gosub CUP {place cup back} 12. gosub TONG {get tongs} 13. gosub TONG1 {get noodles} 14. gosub TONGB {place tongs back} 15. gosub BOWL1 {get bowl} 16. gosub OIL {get oil bottle} 17. gosub OIL2 {add oil to noodles} 18. gosub OILB {place oil bottle back} 19. gosub TASTE {get taste bottle} 20. gosub POPO 21. gosub TASTE1 {place taste bottle back} 22. gosub TONG 23. gosub STIR 24. gosub TONGB 25. gosub POT1 {transfer noodles from pot to bowl} 26. signal -5,,,,,, 27. gosub BOWL2 {transfer bowl to service table and return to SAFE!} . end* ". It must be noted that such a program is PKS-specific in as much as it utilises the tools, processing equipment and location coordinates specific to the PKS in use. However, the functional objectives of the sub-routines are universal since they are required in any PKS provided that the sub-routines are rewritten to suit the environment of the PKS. Thus, it appears feasible to develop recipe programs in a PKS-independent environment and then post-process them to suit the particular PKS in use.

4.THE PKS-INDEPENDENT OFFLINE SYSTEM (PKS-IOS)

PKS-IOS endeavours to create a user friendly programming environment in which recipe programs could be developed from recipes using English-like syntax without worrying about the technical intricacies of the layout, tooling, equipment and location coordinates of the PKS in which the program might be executed. The translation of the program for use in a specific PKS environment is achieved separately by the use of an

appropriate post-processor. The current version of PKS-IOS, which is written in PASCAL for use on a IBM PC Compatible, includes the post-processor for the Experimental PKS. The main menu of PKS-IOS contains options for CREATING, EDITING, SAVING and LOADING recipe programs in addition to a HELP option.

The command syntax of PKS-IOS consists of a KEY-WORD followed by upto four arguments. The key-word is either an action-verb (e.g. GET and ADD) or a REPEAT/TIMING command. The PKS-IOS processor has the ability to interpret the key-word entered by the programmer and prompt him to enter the necessary arguments, viz OBJECT, DESTINATION, STARTING - PLACE and NO. OF CYCLES/TIME-PERIOD. It should be noted that 1 cycle of ADD operation invOVES the dispensation of 1 BU of the material (OBJECT) entered. A 'TIMER.... STOP-TIMING' command sets the duration of the inset command statements in 'minutes & seconds' expressed as integer numbers. A 'REPEAT....FOR' command repeats the inset command statements for the specified number of cycles. Examples of PKS-1 commands are *GET WATER 4 cycles, PLACE SAUSAGE TO PLATE from P for1 cycle, STIR NOODLE at POT for 10 cycles, SWITCH-OFF MICROWAVE OVEN*, and *TIME for 2 min 40 sec for the following action PUSH-FRY SAUSAGE at PAN for 4 cycles WAIT for 0 min 3 sec STOP-TIMING.*

PKS-IOS incorporates an error checking facility where syntax checking is done on line. However, invalid data are checked only during post-processing where, if a VALI level sub-routine corresponding to a command statement does not exist, the system returns the 'No such sub-routine' message. Errors in logic (such as trying to ADD water to the rice-cooker while its lid has not yet been OPENed) cannot of course be checked et> system.

The following is an example of a PKS-independent recipe program developed using the PKS-IOS and, thus, taking advantage of the English-like syntax to avoid the need for prior knowledge concerning the technical intricacies of the PKS in which it might be executed : " *1. OPEN COOKER 2. GET GLASS for 1 cycle 3. REPEAT 4. GET WATER for 4 cycles 5. ADD WATER to POT for 1 cycle 6. PUT BACK GLASS 7.ADD RICE to COOKER for 1 cycle 8. CLOSE COOKER 9. SWITCH-ON COOKER 10. END.*

After completing a CREATE or EDIT session, the user may invoke the post-procesor in order to translate the PKS-independent program into a PKS-dependent program consisting of VALI level sub-routines of the Experimental PKS. The post-processor then decodes each command statement in sequence by recognising the key-word and identifying the corresponding category of VALI sub-routines. The database file belonging to the category is then opened and the appropriate sub-routines extracted. The PKS-dependent recipe program is then output as the concatenation of the VAL1 level sub-routines extracted.

PKS-IOS has been tested by producing a range of recipe programs including boiled rice, baked chicken wing, fried sausage, fried hamburger, etc. The post-processed recipe programs were then executed in the Experimental PKS without further human intervention. The resulting dishes were tasted by a group of lay testers (including a housewife) who judged them as 'edible and enjoyable'.

5. CONCLUDING REMARKS

The work on the Experimental PKS has shown that flexibly automated 'open loop

cooking is feasible with current technology and capable of producing a fairly large range of dishes of enjoyable quality. The major strategies contributing to the technical feasibility are (i) the stipulation of fixed storage locations for all tools and processing equipment, (ii) the utilisation, wherever possible, of the concept of Basic Units for the dispensation of auxiliary materials, (iii) the scientific determination of the process parameters for the process operations associated with each dish, (iv) the execution of all robot motions between predetermined safe points and (v) the adoption of structured programming through the utilisation of canned robot cycles corresponding to the basic robotic operations that are commonly found to a range of dishes.

Further, the English-like syntax of PKS-IOS enables the quick creation of recipe programs without being concerned with the technical intricacies of the particular PKS on which the program might be executed. This feature facilitates the future development of software houses specialising in the writing of recipe programs while allowing the emergence of a variety of Home PKSs as flexible automation in general and PKS technology in particular become increasingly mature and commercially attractive.

A major problem still confronting the development of Home PKS concerns the dispensation and presentation of principal materials. There appears to be no simple technical solution to this problem. Perhaps the answer is in the development of a feeder industry that pre-packages principal materials to suit programmable kitchen systems. Likewise, further work is needed to develop sensors appropriate to cooking and recipe program writing in natural language.

6. ACKNOWLEDGEMENT

The author ackonowledges the contributions made by his students W.L. Lau, B.Y. Ho, Y.H. Wong, H.S. Pang, C.Y. Wong and W.K. Tsui in the development of the Home PKS at the Hong Kong Polytechnic.

7. REFERENCES

1 R.U. Ayres, Robotics, edited by M. Minsky, New York, AnchorPress, 236-263, 1985.
2. P.E. Agre, Robotics, edited by M. Minsky, New York, Anchor Press, 70-97, 1985.
3 I. Asimov and K.A. Frenkel, Robots : Machines in Man's Image, New York, Harmony Books, 44, 1985.
4 E.L. Safford Jr, Handbook of Advanced Robotics, Blue Ridge Summit, TAB Books Inc., 50-58, 1982.
5. Michael Valenti, Mechanical Engineering, 114, January 1991.