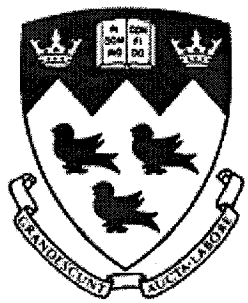
AUTOMATED CONTROL OF THE ELECTRON MICROSCOPY PROTEOMIC ORGANELLAR PREPARATION ROBOT

Punwani, Karishma

Department of Biomedical Engineering
McGill University, Montréal
Canada

October 2005



A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Engineering

Your file Votre référence
ISBN: 978-0-494-25009-9
Our file Notre référence
ISBN: 978-0-494-25009-9

Acknowledgments

I would like to express my gratitude towards my supervisor, Dr. Robert E. Kearney for believing in me and giving me the opportunity to work on such an interesting project. Thank you for your guidance and constant encouragement, they were especially welcomed during challenging times.

I would also like to thank my colleague, Raymond Waterbury, for his kindness, patience, support, and willingness to explain. More importantly, thank you for finally making me realize that with hard work anything is possible. What you have accomplished and done all by yourself will never cease to amaze me.

To my two closest friends, I owe the completion of this thesis to both of you. Thank you for motivating me and for being a constant source of entertainment while I was writing. Moreover, thank you for taking the time to read and re-read all revisions of this thesis.

Lastly, I would like to thank my parents, especially my father, for encouraging me to pursue my Master's and for having faith in me even when I lacked it in myself. This thesis is dedicated to both of you.

This work was supported by funding provided by the Natural Sciences and Engineering Research Council of Canada (NSERC), Genome Québec (GQ), Genome Canada (GC) and the Canadian Institutes of Health Research (CIHR).

Abstract

Electron microscopy (EM) is an important tool in organellar proteomics, where it is used to validate sample purity and to confirm protein presence. Current sample preparation techniques are manual, labor-intensive and time-consuming. To overcome these problems, an electron microscopy proteomic organellar preparation (EMPOP) robot is being developed for parallel preparation of up to 96 subcellular fraction samples in an efficient, repeatable and standardized manner. This thesis describes the development and validation of the software that controls the EMPOP robot. The software was organized in two coordinated levels consisting of a: (1) human-machine interface (HMI), and (2) low-level real-time control routines. The HMI was designed to be 'friendly and flexible', and to enable the operator to modify system parameters on-the-fly. Contrarily, the low-level control routines are responsible for controlling all EMPOP system processes. Pilot studies using the EMPOP system prove that the robot and software function predictably and consistently to generate high quality subcellular sample fractions.

Résumé

La Microscopie Électronique est un outil important dans l'approche par organites cellulaires en protéomique où est utilisée pour valider la pureté des échantillons et confirmer la présence des protéines. Les méthodes de préparation actuelles sont manuelles, nécessitant énormément de main-d'œuvre et de temps. Pour surmonter ces difficultés, un robot à Microscopie Électronique préparateur d'organites cellulaires en protéomique (MEPOCP) est développé pour la préparation parallèle jusqu'à 96 échantillons de fractions sous-cellulaires d'une façon efficace, répétitive et normalisée. Cette thèse décrit le développement et la validation d'un logiciel qui contrôle le robot MEPOCP. Le logiciel a été organisé à deux niveaux cordonnés qui sont : (1) une interface homme-machine (IHM), et (2) des routines de contrôle à temps réel de bas niveau. Il a été conçu pour être 'simple et flexible' afin de permettre à l'opérateur de modifier les paramètres du système à la volée. En outre, il gère la séquence d'exécution des modules de bas niveau. Les résultats d'essais ont démontré que le robot MEPOCP fonctionne de manière prévisible et consistant pour produire des échantillons de fractions sous-cellulaires de haute qualité.

Table of Contents

Table of Figures

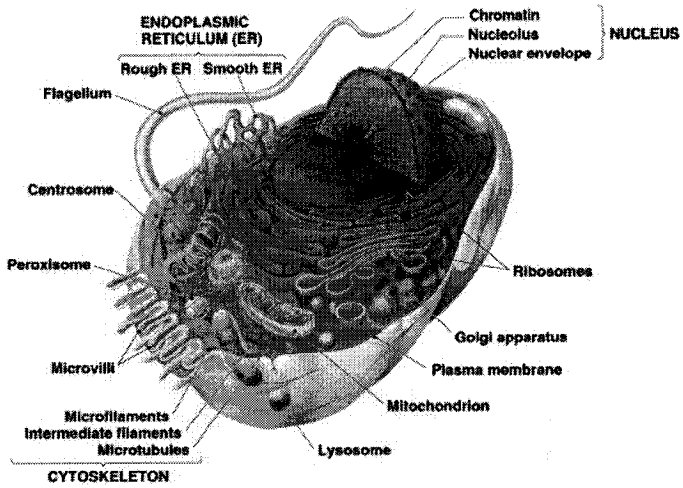
Table of Tables

1 Introduction

In 1663, Robert Hooke made a discovery that revolutionized the world of cell biology; he identified the smallest indivisible unit of life - the cell[1]. Since then, the field of molecular and cell biology has matured and with the advent of new technology we have a greater understanding of cellular function and organization. The cell is responsible for the simultaneous coordination and execution of countless biological processes. These processes are both internal and external to the cell and include mechanisms for cell maintenance and division as well as ensuring proper tissue function via cell signaling and metabolic transportation[2]. A complex, highly organized network of structures called organelles allows the eukaryotic cell to perform its many roles. It is helpful to view organelles as membrane-bound containers, which allow a variety of necessary but incompatible chemical reactions and pathways to occur in isolation thus preventing possible cellular damage[3]. Figure 1.1 and Table 1.1, respectively, illustrate the structure of a eukaryotic cell and describe the function of its more prominent organelles.

Cells and organelles are made up of varying quantities of different molecules including nucleic acids, carbohydrates, lipids and protein macromolecules. Of these, the proteins are central to proper cellular and organellar function and structure[4]. Proteins catalyze organelle specific reactions and provide them with structural stability. In addition, they deliver molecules to and remove waste products from the organelle via surface markers found on its membrane[5]. The cell may be compared to a corporation where proteins act as workers and organelles play the role of managers.

A



B

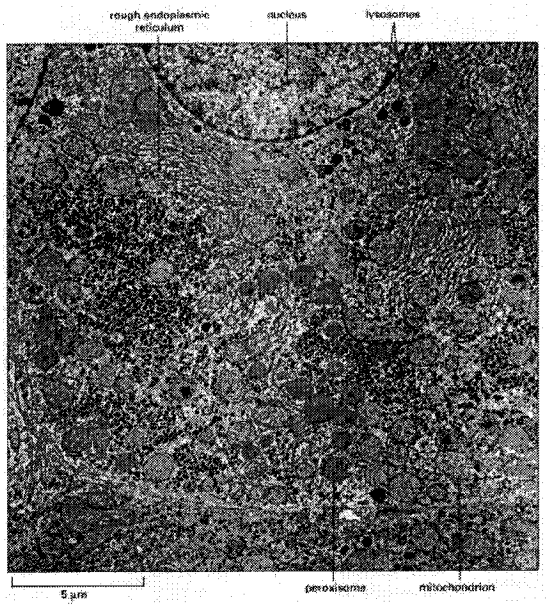


Figure 1.1: Eukaryote cell structure: (A) interpretive drawing[6], (B) transmission electron micrograph[5]

ORGANELLE	FUNCTION
Nucleus	- Stores genetic information - Site for DNA and RNA synthesis
Endoplasmic Reticulum (ER)	- Synthesizes soluble and integral membrane proteins - Transports proteins to other organelles or out of the cell
Mitochondria	- Generates ATP (energy) for cell reactions to occur
Lysosomes	- Breaks down macromolecules - Destroys malfunctioning organelles and unwanted foreign particles taken from outside the cell by endocytosis
Golgi apparatus	- Modifies lipids and proteins - Transports lipids and proteins from the ER to their destined intracellular locations

Table 1.1: Organelle function[5]

Since proteins play such a significant role in cellular function, it is startling to discover that they are built within the cell. In fact, what is more surprising is that the information needed to construct a protein molecule is located within genes, segments of DNA, found in the nucleus of the cell. Furthermore, protein formation does not occur at random; instead, the cell dictates the moment and the place where the macromolecule will be synthesized. It is estimated that the body produces more than a million different proteins[7,8]. Consequently, proteins are generally categorized by their biological functions: (1) enzymes, (2) transport proteins, (3) structural proteins, (4) storage proteins, (5) motor proteins, (6) signaling proteins, (7) receptor proteins and (8) gene regulatory proteins[9]. Proteins are three-dimensional organic compounds made up of long folded strands of amino acids joined by peptide bonds. Protein function arises as a direct result of the 3D structure, which stems from the interaction of attractive and repulsive forces that exist between each amino acid forming the protein chain[10].

The completion of the human genome project (HGP) in 2003[11] left researchers with many unanswered questions regarding cellular function. In addition, the scientific community quickly realized that while mapping the human genome was vital for determining the correlation between genes and proteins it gave limited insight into the function and structure of proteins. The limitations and questions that arose from mapping the human genome gave rise to proteomics. The term proteomics refers to protein identification and quantification, under normal and perturbed states, with emphasis on determining protein location, modification and interaction[12].

At first glance, mapping the human proteome would seem to be easier in comparison to the human genome since only ~2%[11,13] of DNA codes for proteins; however, research has shown that one gene can code for several proteins each with a different structure and function[13,14]. Furthermore, the interpretation of the cell's genetic information can cause a single gene to produce different forms of the same protein due to a plethora of post-translation modifications and alternative gene splicing[15,16].

As a result, mapping the human proteome is a project several orders more complicated than the genome. Furthermore, unlike genes, proteins are modified constantly by changes in their cellular environment. Moreover, protein function is heavily dependent on tissue type. A study conducted in 2003 showed that mitochondrial proteins perform different tasks in the liver than in the kidney[14].

Initially the lack of high-throughput technology hampered proteomics because it prevented researchers from analyzing proteins en-masse. Thus, proteomics in its early stages required an experienced protein chemist to study proteins on an individual basis[13]. However, this changed with the introduction of breakthrough automation technologies based on mass-spectrometry (MS). The advantages of high-throughput proteomics are numerous and include shorter analysis times and the ability to reliably reproduce results in a standardized manner[17].

Experimental approaches to protein analysis vary greatly depending on the protein character under focus[18]. MS based proteomics is the most widely used method to identify and sequence proteins. When this method was first developed researchers hoped to use it as a tool to identify all the proteins within a cell. Unfortunately, this proved to be an overly ambitious goal because it was not possible to isolate low-abundance cellular proteins for examination[19]. In addition, they were unable to gather further functional insight from previously identified proteins. These setbacks resulted in the establishment of organellar or subcellular proteomics, which attempts to identify proteins at the organellar level as opposed to the cellular level. The main advantage of the subcellular proteomics method, over the conventional method, is the determination of a protein's cellular location. Moreover, it also increases the dynamic range and permits the identification of lower abundant proteins. In fact, subcellular proteomics is so powerful that researchers can gain insight on cell function regardless of whether a protein is novel or known. In the first instance, the function of novel proteins can be inferred from previous knowledge of organelles and their functions[14]. Alternatively, the identification of proteins with known functions can be used to enable researchers to develop more information on organellar function.

A key tool in subcellular proteomic research is the transmission electron microscopy (TEM). TEM is an imaging tool, which has been available to biologists since the 1950s; it allows researchers to explore the cell at the structural level[20]. Within proteomics, it is used to determine the homogeneity of organellar/protein samples and to identify the location of novel proteins[21]. The major pitfall of TEM is that it can take several days for an experienced technician to prepare samples for electron microscopic analysis. In addition, the samples are extremely fragile and mistakes made during the preparation stages can render the sample unusable[20].

The challenges faced by laboratory technicians in preparing viable EM subcellular fraction samples for proteomic research have created a demand for a high-throughput laboratory device to automate sample preparation. In response to this demand and through aid from various funding sources, my colleague and I have designed and automated the first-ever, standalone, robot for TEM sample preparation. This system, known as the Electron Microscopy Proteomic Organellar Preparation (EMPOP) robot, will standardize the preparation of up to 96 subcellular fraction samples within the period of a day with limited user interaction. We believe it has the potential to assist the field of organellar proteomics by providing it with an indispensable diagnostic tool.

My role in the development of the EMPOP system was to develop the system for automation and control. It required devising algorithms and programming schemes to serve as the robot's brain and allow it to make operational decisions based upon its senses. Apart from writing programs to control the interaction and integration of all the system's hardware components it was equally important to create a machine interface that would enable the operator to interact with the robot easily. This thesis will describe the programming methodology adopted for the EMPOP robot, provide an in-depth discussion of the algorithms used, and demonstrate their operation.

This thesis is organized into six chapters, including the introduction.

- Chapter Two takes the reader on a tour of MS-based organellar proteomics with a special focus on the use of TEM. This is followed by an in-depth look at the challenges that face proteomists in preparing biological samples for TEM and how researchers came to choose automation as a solution to overcome these problems.

- Chapter Three will familiarize the reader with the robot's overall hardware make-up and illustrate its function during a typical sample preparation run.

- Chapter Four begins with a description of the robot's two-tier control architecture and proceeds with an in-depth look at the software methodologies implemented within each tier. It will then establish how the operator interacts with the machine via the human-machine interface (HMI) and continue to describe the embedded algorithms used to control all robotic processes.

- Chapter Five will explain the methods used to validate the automated control and demonstrate the system's ability to prepare samples.

- Chapter Six concludes the thesis with an overall discussion and suggestions for improvements and future work.

2 Background

2.1 Chapter Overview

This chapter begins by familiarizing the reader with the MS-based organellar proteomics methodology and by exploring the challenges associated with using it as a tool for proteomics research. Moreover, it will explain how researchers apply TEM to verify the purity and homogeneity of protein specimens. It continues by investigating the main bottleneck in using the transmission electron microscope, namely sample preparation. The chapter concludes with the reasons for choosing automation to prepare samples for TEM examination.

2.2 Organellar Proteomics

The goal of organellar proteomics is to determine, locate and classify both known and novel protein species. It aims to gain insight into the function and mechanism of proteins within an organelle by using MS, which deconstructs proteins into their unique amino acid sequences[14]. A typical organellar proteomic experiment (Fig. 2.1) consists of three stages: (1) protein extraction, (2) protein/peptide separation, and (3) mass spectrometric analysis.

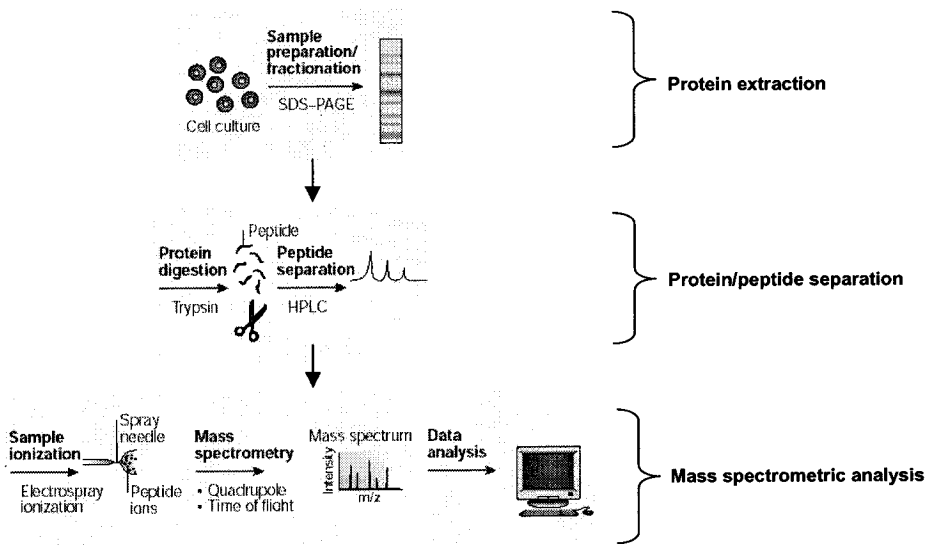


Figure 2.1: Organellar proteomic experimental pipeline[22]

Protein extraction

Subcellular fractionation is a complex cellular processing method used by researchers to acquire organellar protein samples. The technique involves removing and separating organelles from the cell and occurs in two steps: (1) homogenization and (2) fractionation[23]. Cell homogenization refers to the disruption and decomposition of the cell membrane to liberate organelles. Assuming perfect homogenization, the result, known as the homogenate, is a viscous slurry containing thousands of intact, undamaged organelles of distinct size, charge and density[23,24]. Cell fractionation is the technique used to isolate a particular organelle from the homogenate. The most common procedure employs centrifugation to separate cellular organelles based on either their weight or buoyancy[23,25]. Fractionation is generally repeated several times in order to enable researchers to obtain rich, uncorrupted organellar specimens.

The next step after fractionation consists of rupturing the organelle's membranes to release its proteins. A typical organellar specimen contains thousands of different proteins; for example, investigation of the human mitochondrion suggests the existence of at least 1500 distinct proteins[16]. SDS-PAGE is the most common method that researchers employ to separate proteins. As shown in Figure 2.2, SDS-PAGE is a gel technique, which works by ionizing proteins with negatively charged molecules of sodium dodecyl sulphate (SDS); this results in a negatively charged SDS-protein complex. The complex is then loaded onto a polyacrylamide gel where an electric field causes the protein macromolecules to migrate down the gel forming bands. The rate of migration for a protein is directly related to its mass, and hence its ionization; smaller protein molecules migrate further down the gel than larger molecules[23]. Unfortunately, separation is far from perfect since proteins with similar mass and charge will inevitably migrate to the same location.

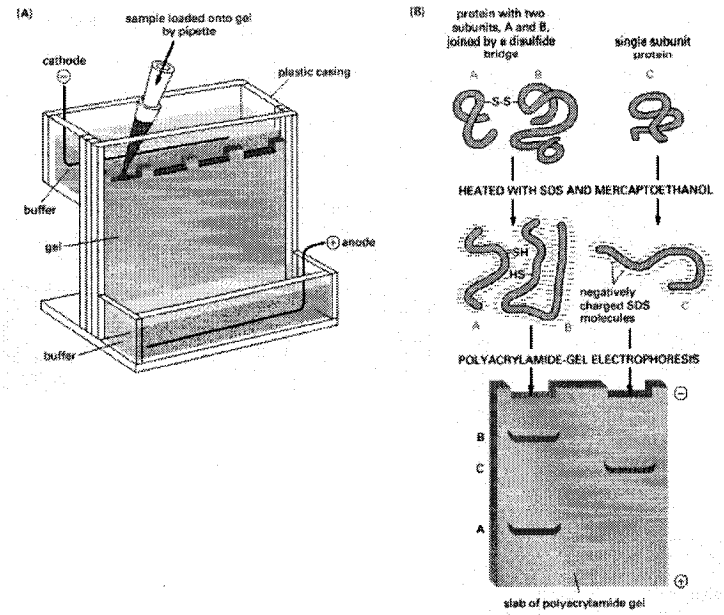


Figure 2.2: SDS-PAGE technique: (A) apparatus, (B) negatively-charged protein complex migrating down gel[23]

Protein peptide separation

After gel separation, protein analysis continues by excising the gel to remove the bands containing the proteins of interest. Although the mass spectrometer can ascertain the mass of intact proteins, doing so gives researchers little insight into its identification since many proteins have the same mass. Moreover, the mass spectrometer yields better, more detailed results when sequencing proteins having an amino acid composition of ~20 residues due to its finite resolution. Consequently, protein macromolecules are digested into small peptide fragments of varying lengths prior to entering the mass spectrometer[26]. Digestion is achieved by treating the molecule with a sequence specific protease, such as trypsin, which causes the protein's peptide bonds to sever at specific, known locations[22].

Unfortunately, due to their number, the resulting peptide fragments cannot collectively enter the mass-spectrometer. Consequently, they are staggered into the mass-spectrometer, in order of increasing hydrophobicity, via a high performance liquid chromatography (HPLC) column[22]. The HPLC column applies a pressure gradient across a steel cylinder column, containing a tightly packed, porous, solid matrix of beads having exposed hydrophobic side chains. The interaction of the protein molecules with

the hydrophobic beads cause the peptide molecules to progress through the column at different rates[23].

Mass spectrometric analysis

Figure 2.3 illustrates the first two steps of the mass spectrometric analysis stage, namely sample ionization and mass spectrometry. Although the depiction suggests that the column and mass spectrometer are two separate units, in reality, the column mounts directly to the mass spectrometer. As the peptide fragments reach the end terminal of the HPLC column they are injected into the mass spectrometer using an ionization technique known as electrospray ionization (ESI). ESI works by forcing the peptides through a small nozzle in the presence of an electric field (typically ~2 kV) which results in a fine spray of highly charged droplets. Once inside, electric fields within the mass spectrometer accelerate the protonated peptide ions towards a detector[24].

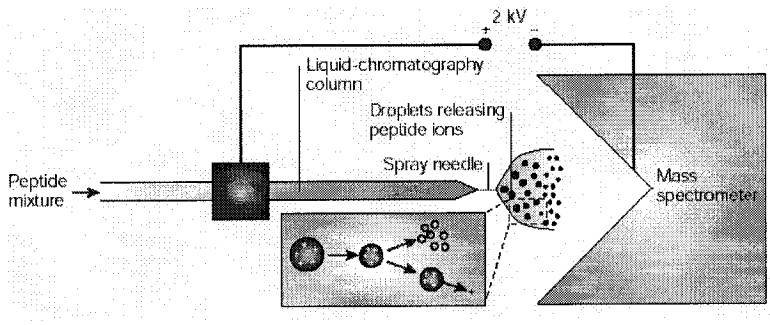


Figure 2.3: HPLC column using ESI to direct peptides into the mass spectrometer[22]

The output of the mass spectrometer (Fig. 2.4A) is a discrete curve, termed mass spectrum, representing the number of ions with a particular mass-to-charge (m/z) ratio. The mass spectrometer generates these m/z ratios by measuring the time it takes for each peptide fragment to reach a detector. This information is then presented to a protein database, such as Genbank or SwissProt, which attempts to find the gene that encodes for the protein with that particular mass spectrum profile[23,27]. Alternatively, the output of the mass spectrometer can be manipulated to determine a peptide's amino acid constituents (Fig. 2.4B)[22]. This requires the mass spectrometer to further cleave the peptide molecule into several fragments; each fragment consists of an additional amino acid residue. The detector measures the m/z for each fragment and outputs a unique

fragment ion spectrum, which is subsequently presented to a database for amino acid identification.

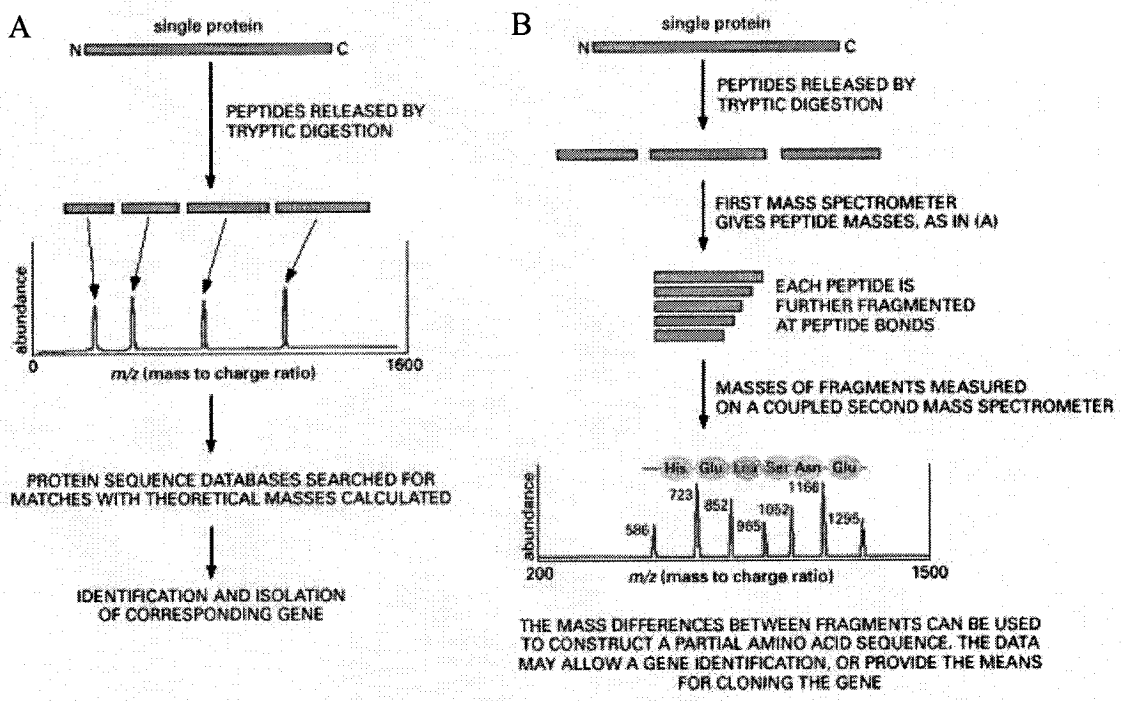


Figure 2.4: Mass spectrometric analysis methods and outcomes: (A) peptide mass spectrum, and (B) its amino acid sequence[23]

2.2.1 Limitations of organellar proteomics

Organellar proteomics is not without its flaws. Problems plague each of the three experimental stages, making it difficult for researchers to interpret their findings and doubtful of their conclusions. A major impediment encountered by proteomic scientists occurs during sample preparation and stems from the inability to obtain pure organelle samples. Errors at this stage carry through to subsequent steps, creating a snowball effect. As a result, if scientists are unable to guarantee the purity of the starting sample, they cannot, with 100% certainty, verify the location and infer the function of an unknown protein[28].

One of the first steps that scientists took to overcome the sample preparation problem was to use cultured cells rather than animal-derived ones, as they could better control the

growth and development of these cells. To their dismay, they quickly realized that cultured cells have a tendency to lose up to 50% of their organellar components during cellular homogenization. Moreover, since organelles (especially smooth vesicle organelles) tend to have similar physical properties, cellular fractionation is unable to fully separate organelles. This causes the desired organellar sample to be laden with unwanted, residual species. In addition, even if it were possible to create the ideal homogenate and to isolate its organelles perfectly, centrifugation during homogenization and fractionation may cause permanent, irrevocable damage to the organelles and hence their proteins[28]. Consequently, before undertaking the next stage of proteomic analysis, researchers use quality control methods to detect sample irregularities that might arise during sample preparation. One such method that is gaining prevalence within organellar proteomics is TEM[28].

2.3 Transmission Electron Microscopy

2.3.1 Electron microscope operation

TEM is a valuable imaging tool, which is capable of detecting a high-degree of detail on the structure and composition of a variety of specimens, ranging from cell fragments to iron metal shavings. Simply stated, a transmission electron microscope (Fig. 2.5) is a more sophisticated version of the conventional light microscope. It is capable of achieving a resolution 10 000 times greater than a light microscope and can resolve features down to distances of 0.1 nm[1].

The electron microscope mimics the light microscope in design and function. The machine consists of four main components: (1) an electron source, (2) a specimen holder, (3) electromagnetic coils and (4) a viewing screen. As its name implies, the electron microscope uses electrons that travel with a wavelength of ~0.002 nm to illuminate the specimen. The electrons travel down a vacuum column where electromagnetic coils, dispersed along the length of the microscope, behave as a lens to focus the particles into a single thin beam. The presence of vacuum within the microscope is crucial for beam formation because it prevents the electrons from

randomly scattering as they collide with air molecules. Once it reaches the specimen holder, the electron beam attempts to penetrate the sample. Penetration causes some of the electrons to reflect away from the main electron beam. The amount of reflection or ray scattering is directly dependent on the density of the substance in question. Consequently, only electrons that permeate through the specimen arrive at the phosphorescent screen, to produce a "shadow image" of the specimens form. This density-based image, where dark regions represent specimen areas of high density, is the product of the transmission electron microscope[1,,29].

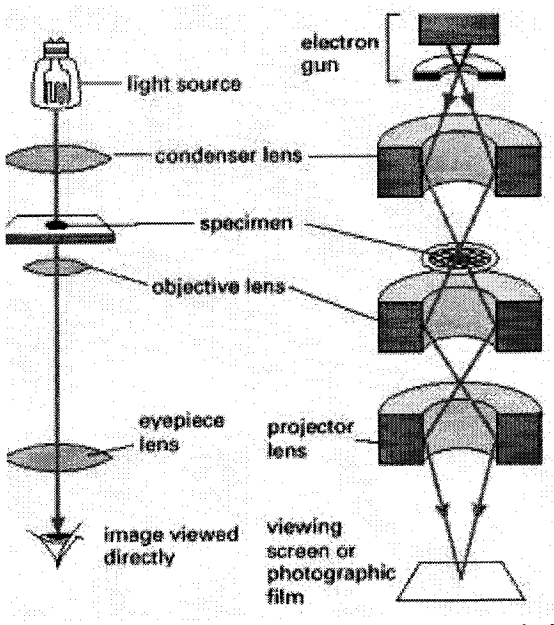


Figure 2.5: Comparison between the light microscope and the transmission electron microscope[1]

2.3.2 TEM within organellar proteomics

As Figure 2.6 shows, TEM is normally used at two stages within the proteomics experimental pipeline. First, researchers employ it during the sample preparation phase to determine quantitatively the quality of a subcellular fraction. Secondly, it is used at the completion of the mass spectrometric analysis experiment as a means to confirm qualitatively, a protein's subcellular location[16,28].

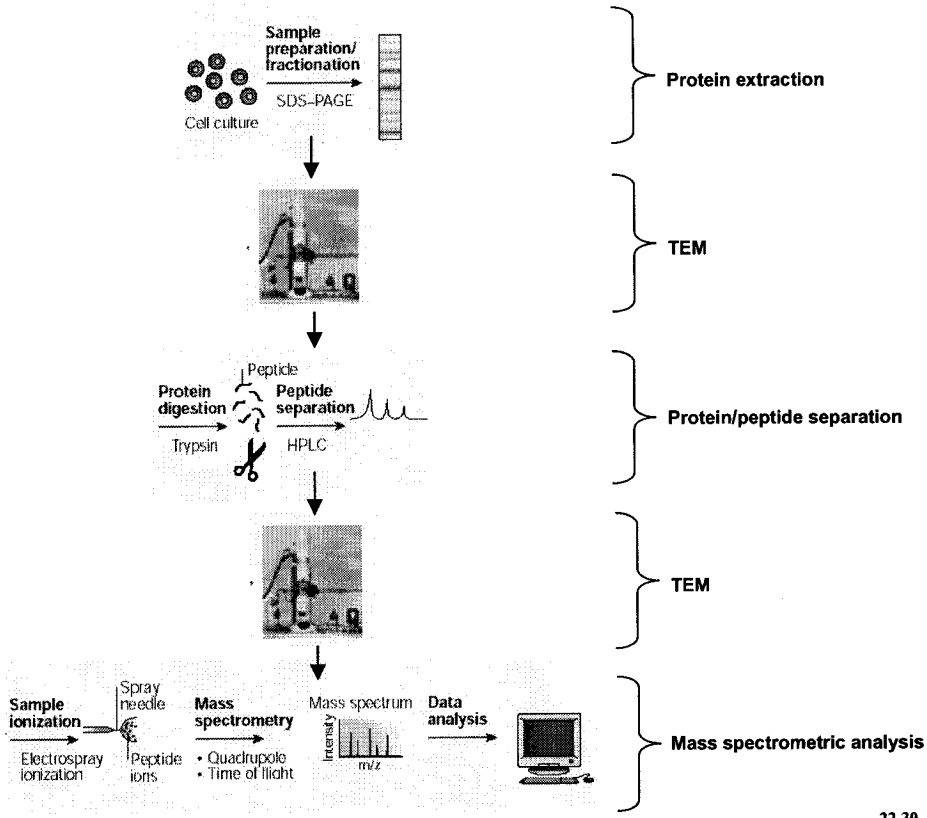


Figure 2.6: Organellar proteomic experimental pipeline including TEM[22,30]

The most widespread method of measuring sample quality is to count the number of desired organelle structures versus the number of contaminants in the fractionated homogenate. This is illustrated in Figure 2.7, which shows how TEM is used to verify the concentration of rough microsomes after ER organelle fractionation. Rough microsomes house more than 20 different types of proteins not found in smooth microsomes[5]. As a result, to further our understanding of how rough ER proteins function, it is vital that the initial organellar sample and subsequent protein fragments originate solely from rough microsomes. In this example, fractionation via density-equilibrium centrifugation was used to separate the rough and smooth ER microsomes. The rough microsomes studded with ribosomes (represented by black dots) sediment below the smooth microsomes (Fig. 2.7A). In practice, separation is imperfect causing the boundary between the two types of organelles to blur, resulting in an impure final rough microsome sample. The type and source of the contaminant(s) present after subcellular fractionation can be verified by micrograph analysis. For ER organelle

fractionation the micrograph reveals the presence of a few smooth microsomes (dashed circle) amongst the mostly abundant rough microsomes (solid circle) sample (Fig. 2.7B).

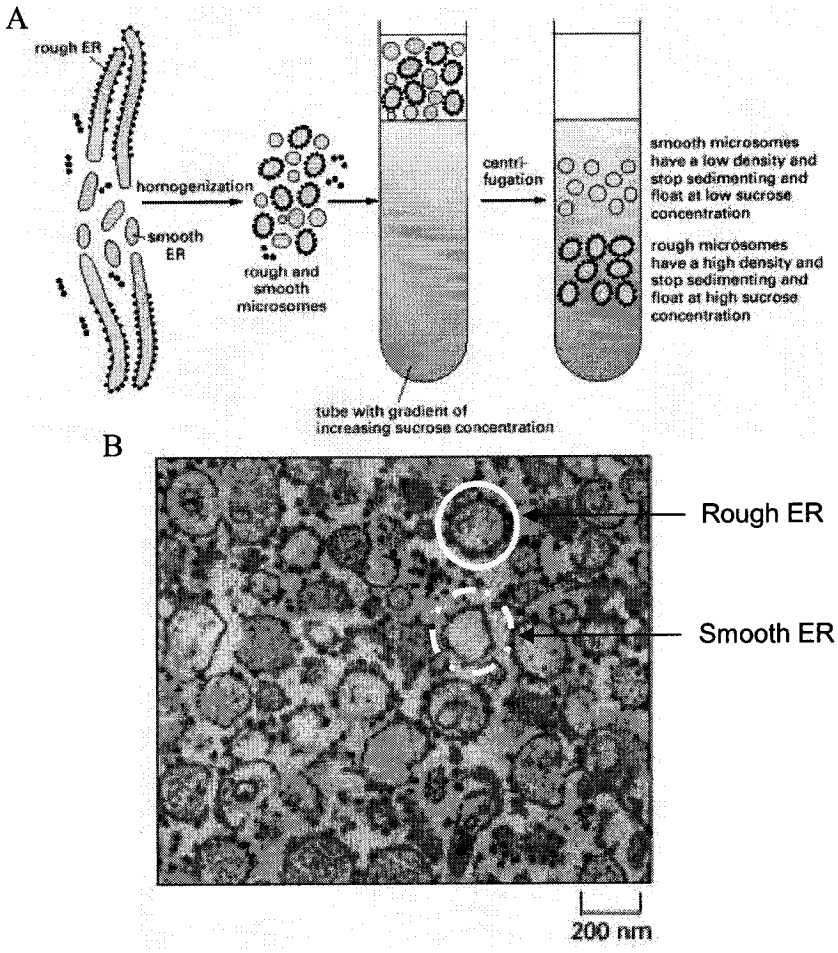


Figure 2.7: Validation of sample purity after subcellular fractionation: (A) separation method, (B) electron micrograph[5]

TEM is also utilized to visualize the location of organellar proteins. However, since the electron microscope can yield only a one dimensional projection of the sample, it cannot directly visualize proteins. This is because a protein molecule in its native state is a three dimensional object[31]. Moreover, unlike other molecules, identification of a particular protein requires information regarding its unique folding pattern, since proteins have varying structures based on their function and location within the cell. Consequently, cell biologists employ immunogold labeling to indirectly ascertain the location of a

particular protein macromolecule. As seen in Figure 2.8, this labeling technique uses antibody-antigen specific targeting to detect proteins[10].

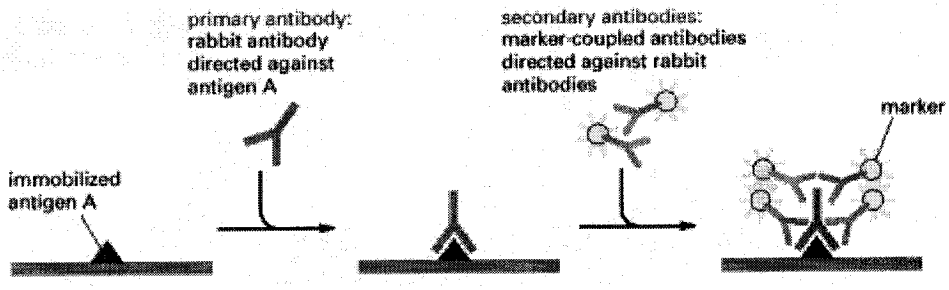


Figure 2.8: Immunogold labeling technique using antibody-antigen protein targeting[1]

It functions by injecting an antigen, the protein molecule under investigation, into a small animal causing a negative immune response. The animal's immune system, via lymphocytes and T-cells, produces antibodies to attack the undesired foreign molecules. After a few days, the animal's spleen is excised to collect the newly formed primary antibodies. This process is then repeated, this time using the primary antibodies acting as antigens, to produce secondary antibodies. After synthesis of both primary and secondary antibodies is complete, immunogold labeling is achieved by incubating the original subcellular sample with primary antibodies causing them to bind to the specific protein species. The secondary antibodies undergo a chemical 'gold-platting' treatment, in which they are covalently coupled to dense gold particles. The last stage of immunogold labeling involves the attachment of the secondary antibodies to the primary antibodies[1,32]. Subsequently, electron microscopic analysis will localize the gold-labeled protein complexes, which appear as small black dots (Fig. 2.9).

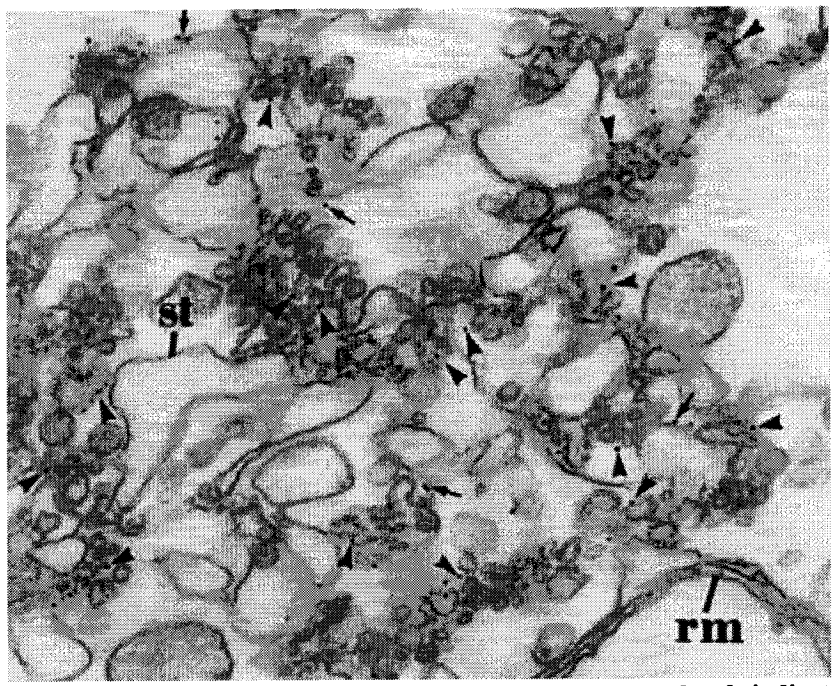


Figure 2.9: Immunogold electron micrograph of the rough ER – arrow heads indicate location of proteins (black dots)[33]

2.4 Sample Preparation

TEM brings many benefits to organellar proteomics; unfortunately, it has one major shortcoming: biological samples must undergo an extensive chemical treatment prior to being micrographed. This is necessary to help protein specimens withstand electron bombardment and the harsh vacuum environment present inside the transmission electron microscope. EM sample preparation, is laborious, tedious, time-consuming and extremely sensitive. Preparation requires a trained individual to ensure that the sample remains preserved in their native state and that sample processing does not introduce reaction artifacts[20,34].

2.4.1 Sample preparation criteria

A sample must meet five stringent prerequisites before TEM can proceed: (1) the sample must be devoid of water, (2) sample integrity must be maintained under high vacuum conditions, (3) the sample must withstand an electron barrage, (4) the sample must consist of electron opaque and transparent regions and (5) the sample must be between 1-10 μm thick[1,20].

Unfortunately, biological samples fulfill none of these requirements. In fact, proteomics samples have the complete opposite properties. Cells consist of almost 70% water and when fractionated the homogenate is placed in an aqueous solution to prevent protein denaturation. In addition, subcellular fraction samples are fragile and structurally weak. Furthermore, they lack areas of varying density and so have poor contrast. Lastly, biological samples are usually much thicker than required for TEM[20].

To overcome these problems and enhance compatibility with TEM, samples are treated chemically. EM sample preparation is highly sensitive, extremely demanding, complex and variable. The most common preparation method involves five main stages: (1) Fixation, (2) Dehydration, (3) Embedding, (4) Sectioning and (5) Staining. Fixation is a preservation technique, which quickly arrests a samples cellular process to capture its "normal" morphology and chemical composition. It includes a method, known as filtration, to secure the sample onto a charged filter membrane. Figure 2.10, is a photograph of the filtration apparatus. It works by using nitrogen gas to push the sample within the cylindrical column onto a nitrocellulose membrane, which traps the sample on its surface. Thereafter, the filter membrane is removed and placed in a test tube where the dehydration, embedding and staining stages are executed. Dehydration is achieved by removing water from the sample via osmosis. Embedding refers to impregnating the sample with a resin, which hardens to support the sample during sectioning. Sectioning involves using an ultramicrotome with a diamond knife, to cut the sample into thin slices between 1-10 μm thick. Staining is the method employed to visualize the sample's structure by coating it with heavy metals, resulting in varying degrees of contrast depending on the macromolecules affinity to a particular metal. Except for staining which can take place either before or after embedding, each stage must follow in sequence and must come to completion before proceeding to the next stage.

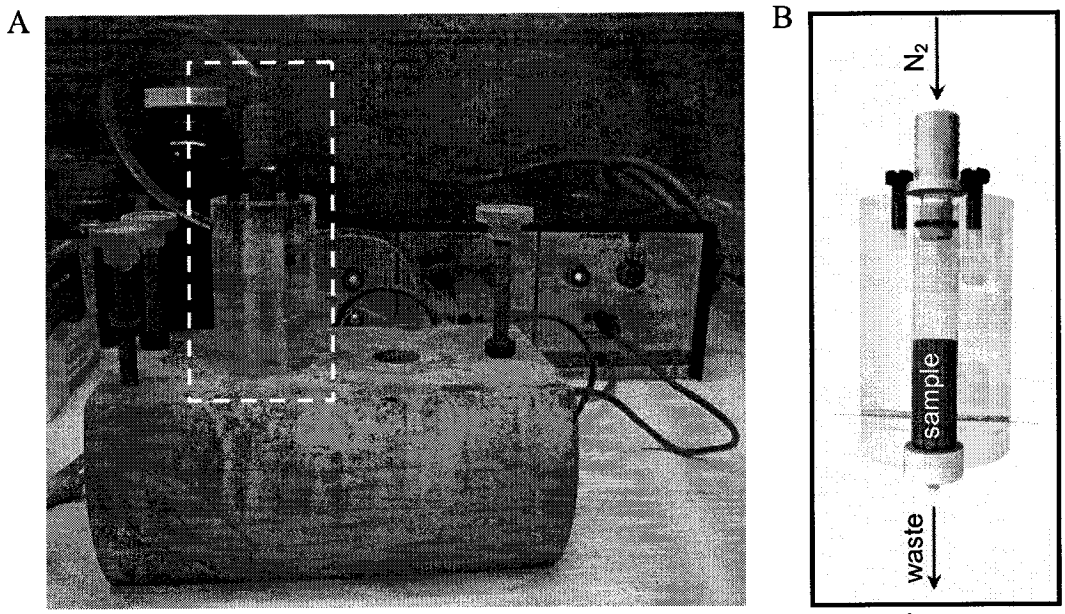


Figure 2.10: Filtration apparatus: (A) photograph, (B) schematic

2.4.2 Current manual protocol

Proteomic researchers at McGill University use either of two manual protocols (refer to protocols on the next page) to prepare samples for EM analysis. Apart from the reagents used during staining, these protocols are similar and include steps for fixation, dehydration and embedding. Researchers select between the protocols depending on which staining method is better able to highlight and discern the protein sample's features in a micrograph.

It is evident that the chemical treatments used for TEM sample preparation are extensive. Moreover, the reagents required are highly toxic in nature. The time and dedication required to prepare samples is large, and technicians must constantly supervise each step in the protocol – making it extremely difficult for them to pursue other work. In addition, EM sample preparation takes approximately four days to complete since technicians work 8-hour days and so must leave their samples to incubate overnight.

Protocol 1

Fixation and filtration:

1. Fixation solution contains: 5% glutaraldehyde, 1mM cacodylate buffer, pH 7.4 and, 0.1% CaCl2
2. Add equal volume of fixative to 2 ml of 100 mg/ml membrane protein, leave in cold O/N
3. Filter on Millipore filter type HA 0.45 mm or 0.8 mm
4. Wash filter 3X with 100 mM cacodylate buffer, pH 7.4 and leave in buffer O/N

Staining:

5. Fix sample in 2% reduced OsO4 in 100 mM cacodylate buffer, pH 7.4 for 1 hr at 4°C
6. Wash 3X with 0.25 M cacodylate buffer, leave O/N
7. Rinse with 0.1m malate buffer, pH 5.7
8. Incubate in 6% uranyl acetate 100 mM malate buffer, pH 5.7 for 2 hr
9. Wash 2X with malate buffer
10. Wash 1X with cacodylate buffer

Dehydration:

11. 50% EtOH for 10 min
12. 70% EtOH for 10 min
13. 90% EtOH for 10 min
14. 95% EtOH for 10 min
15. 100% EtOH 3X for 10 min

Embedding:

16. Propylene oxide for 1 hr
17. Transfer to preembed block
18. 1:3 Epon:propylene oxide for 1 hr
19. 1:1 Epon:propylene oxide for 1 hr
20. 100% Epon for 1 hr
21. Polymerize

Protocol 2

Fixation and filtration:

1. Fixation solution contains: 5% glutaraldehyde, 1mM cacodylate buffer, pH 7.4 and, 0.1% CaCl2
2. Add equal volume of fixative to 2 ml of 100 mg/ml membrane protein, leave in cold O/N
3. Filter on Millipore filter type HA 0.45 mm or 0.8 mm
4. Wash filter 3X with 100 mM cacodylate buffer, pH 7.4 and leave in buffer O/N

Staining:

5. Incubate in 1% tanic acid in cacodylate for 1 hr. with three changes
6. Wash with 1% sodium sulfate in 100 mM cacodylate for 20 min.
7. Incubate in 5% uranyl acetate, 100 mM malate buffer, pH 5.7 for 2 hr
8. Wash 2X with malate buffer
9. Wash 1X with cacodylate buffer.

Dehydration:

10. 50% EtOH for 10 min
11. 70% EtOH for 10 min
12. 90% EtOH for 10 min
13. 95% EtOH for 10 min
14. 100% EtOH 3X for 10 min

Embedding:

15. Propylene oxide for 1 hr
16. Transfer to preembed block
17. 1:3 Epon:propylene oxide for 1 hr
18. 1:1 Epon:propylene oxide for 1 hr
19. 100% Epon for 1 hr
20. Polymerize

* Note: O/N means overnight.

Another problem associated with the manual method is the high inter-rater sample variability. Each technician has their own unique style of preparing samples ranging from their pipetting technique to the accuracy of measuring incubation times. Moreover, technicians do not rigorously follow the protocols. Instead, they treat them as guidelines for preparing EM samples and often adjust the protocol's parameters. In addition, technicians are human and prone to making mistakes, both in the laboratory and in their notebooks. Such mistakes can be devastating since protein samples are precious and scarce. As a result, there is a need for a standardized procedure for EM sample preparation.

2.5 Development of EMPOP Robot

Automation within proteomics has been repeatedly shown to be a highly effective and worthwhile investment. In the case of EM sample preparation, automation may be expected to yield results in shorter times and give researchers the ability to quickly and reliably monitor the quality of cell fragment samples. Additionally, automation would instill researchers with more confidence in their findings and conclusions, by providing them with access to a standardized, repeatable verification tool.

The easiest method for automating EM sample preparation would be to modify current technologies and incorporate them into an automated system. Unfortunately, the majority of laboratory equipment available on the market consists of liquid handlers. Liquid handlers could easily automate the chemical processing aspect of the protocol. However, their use would necessitate the creation and development of devices to handle the first and last few preparation steps, namely sample filtration and transferring of the semi-prepared samples to the preembed block. The interaction of several separate robotic subcomponents would be one possible automation scheme. However, this method is not feasible, since it does not result in an autonomous system; necessitating a technician's presence, a reality we are trying to avoid. Moreover, the constant perturbation experienced by the samples as they are moved to different machines could result in irreversible structural damage to the delicate proteomic samples.

Consequently, this automation approach was abandoned, and it was decided to develop a standalone system, from scratch, to automate all aspects of EM sample preparation. Moreover, in addition to providing standardized, repeatable processing, the EMPOP robot was designed to increase proteomic throughput by simultaneously processing of up to 96 subcellular sample fractions within a single day. As this project has many facets, it was divided between me and another student. The task of conceptualizing, designing and fabricating the mechanical and electrical system was assigned to a PhD student, while I took the responsibility of programming the system for automated control and function, as my master's thesis.

3 Robot Design and Operation

3.1 Chapter Overview

This chapter discusses the mechanical structure of the EMPOP robot, describes its various subsystems, and explains its role in preparing specimens for TEM examination. The robot's conceptualization, design, and fabrication were carried out by Mr. Raymond Waterbury as his PhD project. Consequently, this chapter only provides an overview of the system; a detailed discussion of the hardware design choices and specific operation of each component and circuit are beyond the scope of this thesis. However, a solid understanding of the robot's structure and function is required for the reader to understand the control software.

3.2 Mechanical Design

The EMPOP robot was designed to handle all aspects of sample preparation from fixation, through to embedding in a single device. Moreover, it was built to process up to 96 subcellular sample fractions simultaneously. Figure 3.1 is a photograph of the front of the EMPOP robot. It clearly illustrates the complexity of the mechanical components required to achieve a high-throughput, repeatable, standardized, robust system. The design aimed to create a standalone, autonomous device by integrating seven modular subsystems: (1) a core mechanism, (2) a 5-axis motion control system (X, Y, Z θ, ϕ), (3) a cooling platform, (4) an electromagnetic arm, (5) a plate transfer platform, (6) a pneumatics manifold and (7) a fluids handling mechanism. Each subsystem will be described in detail.

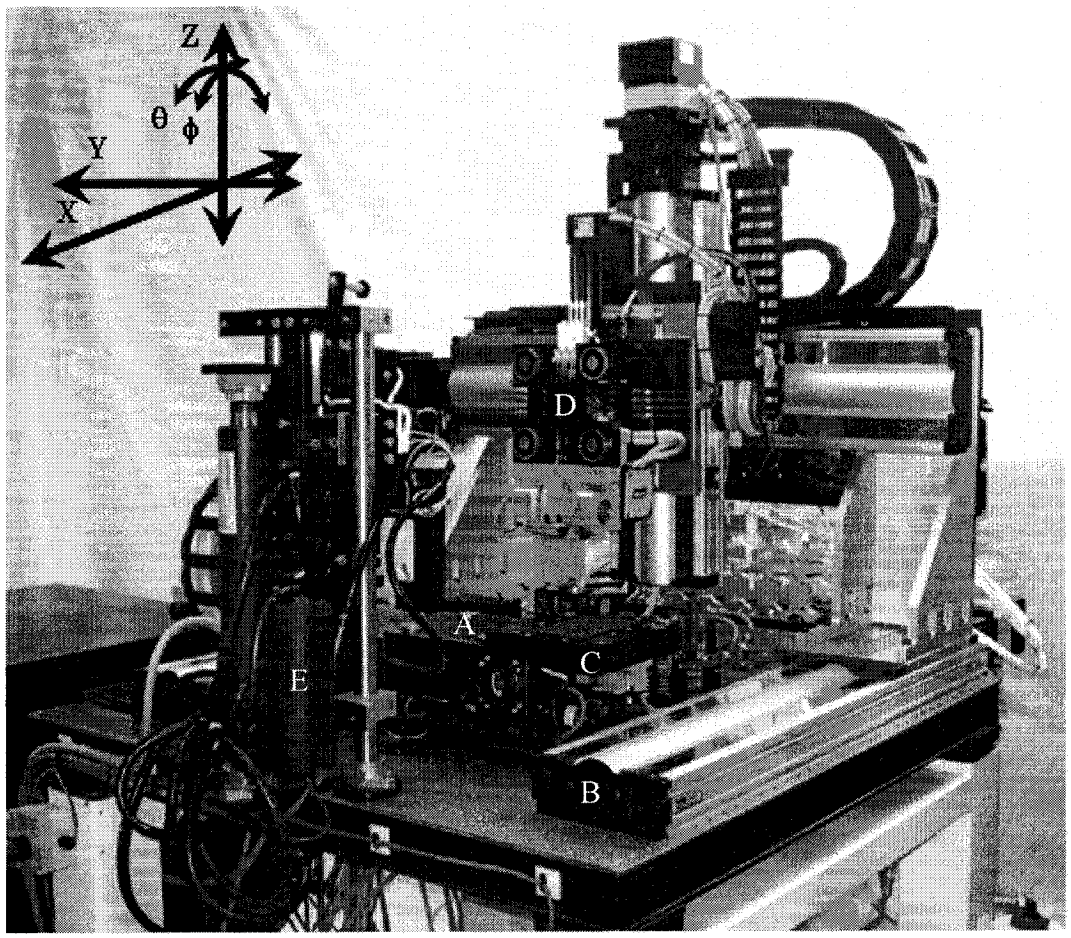


Figure 3.1: Photograph of EMPOP Robot consisting of seven modular units: (A) core mechanism, (B) 5-axis motion control, (C) cooling platform, (D) electromagnetic arm, (E) plate transfer platform, (F) pneumatics system (not visible) and (G) fluids handling subsystem (not visible)

Core mechanism

The core mechanism (Fig. 3.2) is central to robotic sample preparation. It consists of four plates: (1) a fluid transfer plate (FTP), (2) a filter plate, (3) a vacuum/pressure plate (VPP) and (4) a retainer plate. All aspects of subcellular sample processing, including filtration, delivery and removal of chemical processing reagents, and ejection occur within these plates.

- The FTP provides an individual fluid reservoir for up to 96 samples, limiting the risk of cross-contamination between samples

- The filter plate contains a nitrocellulose membrane backed by a filter screen, which acts to secure organellar samples throughout most processing

- The retainer plate holds a 96 well plate of polymerized embedding solution and is the site where embedding occurs

- The VPP has a dual role of supplying a vacuum *pulse* to adhere samples onto the filter membrane, or a pressure burst to retrieve them from it

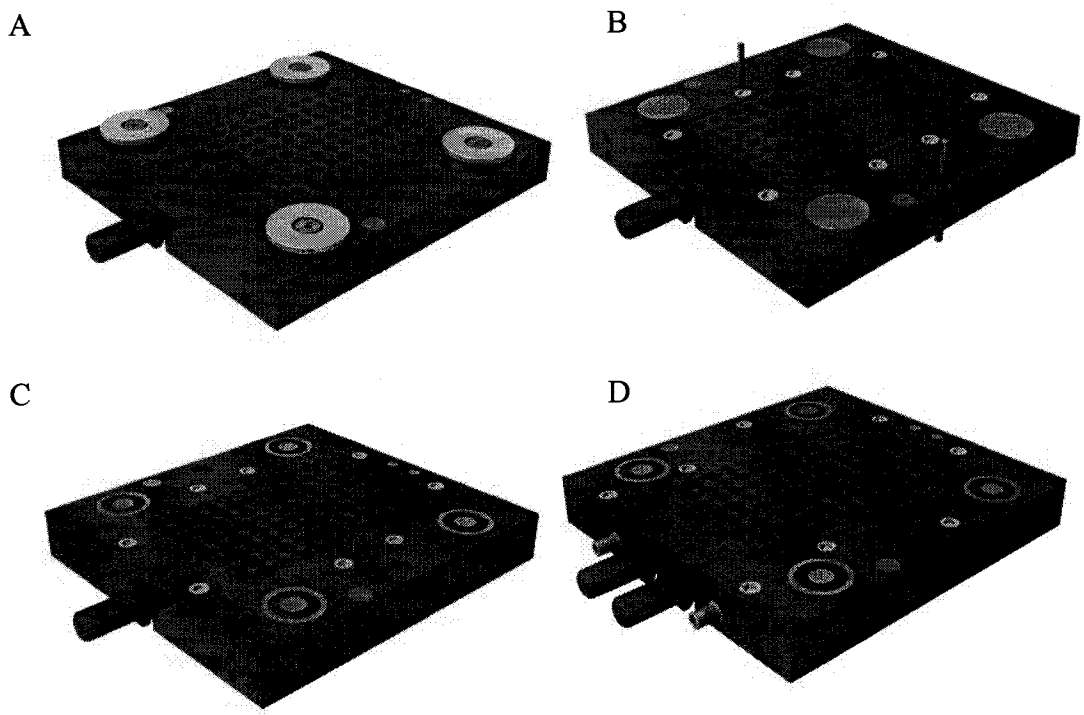


Figure 3.2: Core mechanism plates: (A) Fluid transfer plate (FTP), (B) Filter plate, (C) Retainer plate, (D) Vacuum/Pressure plate (VPP)

During processing, the plates are configured into three different arrangements.

- A processing configuration (Fig. 3.3) comprising of the FTP, filter plate and VPP

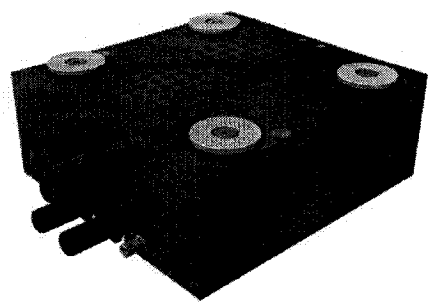


Figure 3.3: Processing configuration

- A delivery configuration (Fig. 3.4) comprising of the VPP, filter plate and retainer plate

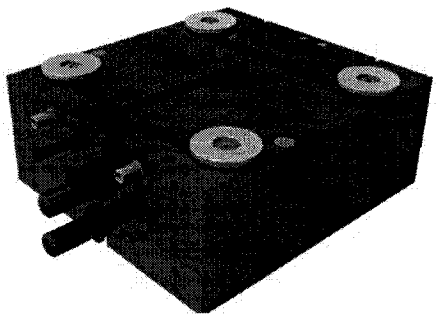


Figure 3.4: Delivery configuration

- An embedding configuration (Fig. 3.5) comprising of the filter plate and retainer plate

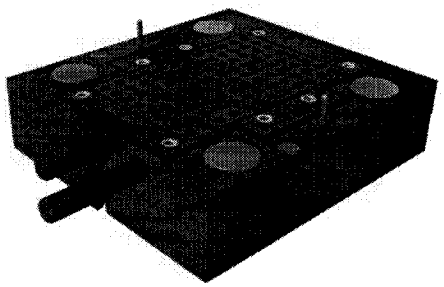


Figure 3.5: Embedding configuration

The processing and delivery configurations (Figs. 3.3 and 3.4) are assemblies of three plates, whereas the embedding configuration has only two. The delivery and embedding plate configurations are used in the last few stages of processing to transfer and embed the proteomic samples in Epon resin.

Figure 3.6, shows an exploded CAD rendering of the processing configuration, where plate compression follows along the arrows. The filter plate is always located at the center of either three configurations and is fitted with eight permanent magnets (four on each face). Electromagnets incorporated on the inner face of the remaining three plates are used to couple and decouple them from the filter plate. Conversely, steels discs attached to the outer face of the FTP, VPP and retainer plate enable the arm to attach to the plate assemblies more readily.

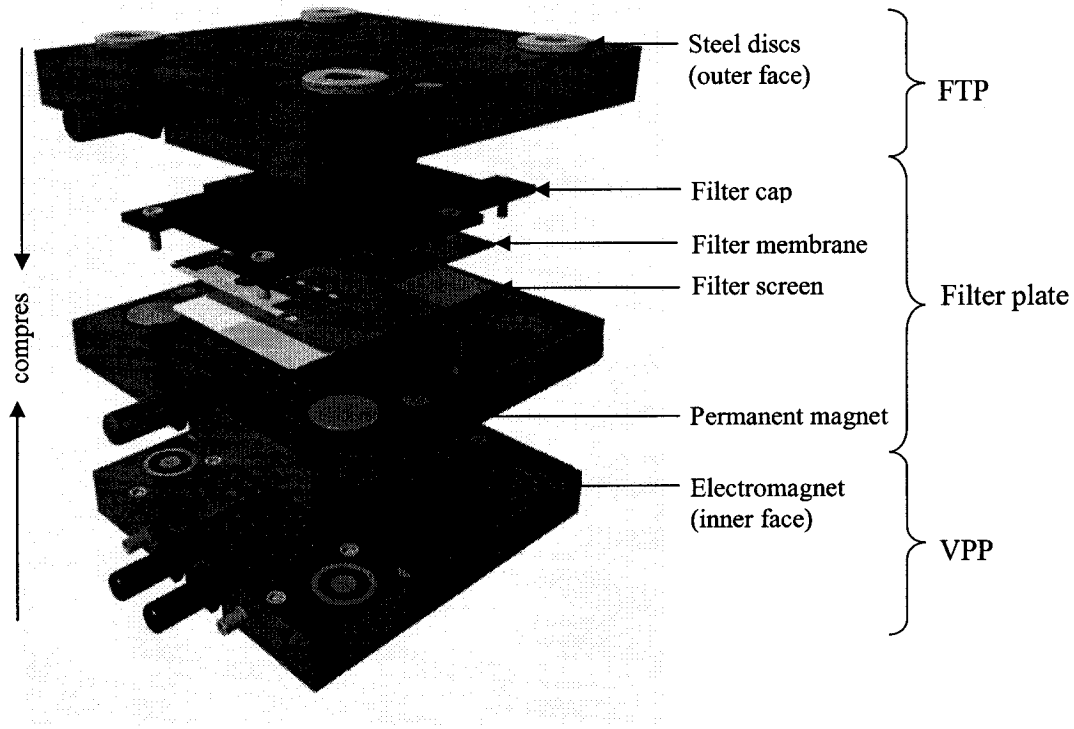


Figure 3.6: Exploded view of processing configuration

Cooling platform

The primary function of the cooling platform (CP) (Figs. 3.7 and 3.8) is to maintain the core mechanism and the samples at a temperature of approximately 4°C. Cooling inhibits the rate of protein denaturation and so preserves the underlying structure and overall chemical integrity of the samples. It is achieved by Peltier junctions, thermo-electrical cooling pads, located underneath the surface of the CP.

The CP also incorporates a mechanism to compress and retract its inner walls. This facilitates plate interaction by providing the system's electromagnetic arm with enough clearance to manipulate the plates. The two elevated surfaces of the CP, surfaces A and B, serve to store decoupled plates. Magnetic sensors implanted within the surfaces detect the presence of plates.

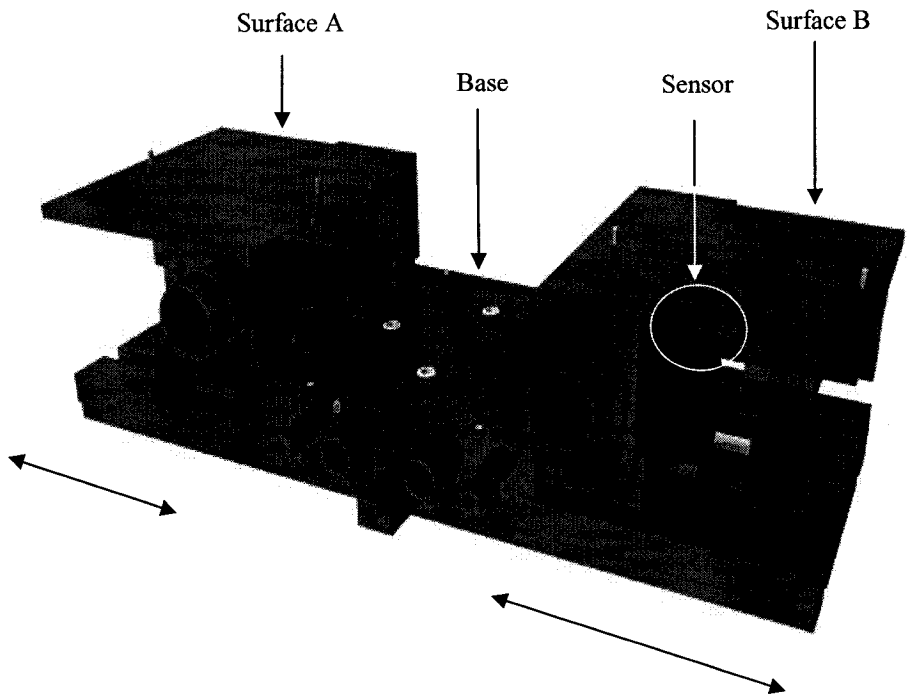


Figure 3.7: Schematic of CP (double-headed arrows indicate the direction of plate compression and retraction)

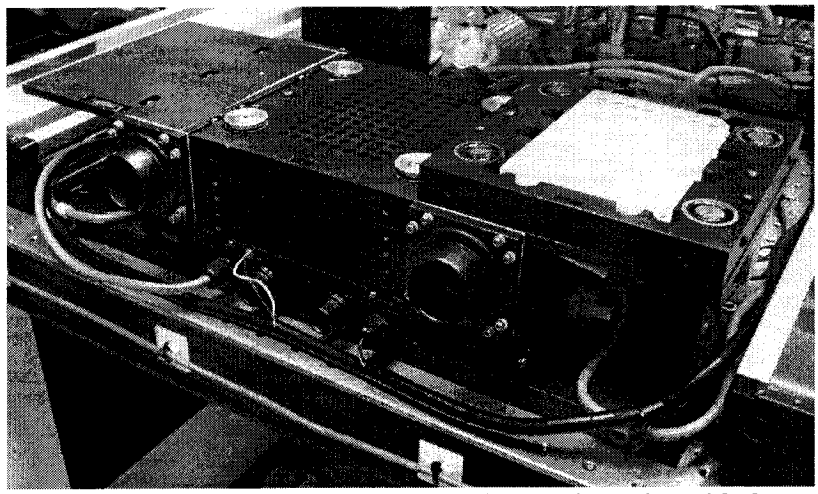


Figure 3.8: Photograph of CP supporting the processing configuration with the retainer plate resting on CP's B surface

5-axis motion control system

The robot's 5-axis (X, Y, Z, θ, ϕ) motion control system is the largest EMPOP subsystem. It is a 5-axis serially coupled system and enables the robot to move to the clean station for needle cleaning, drying and priming and to the core mechanism for

reagent dispensing, aspirating and plate interchanging. Figure 3.9 shows a CAD rendering of the robot's motion control system; Figure 3.10 shows a photograph of it.

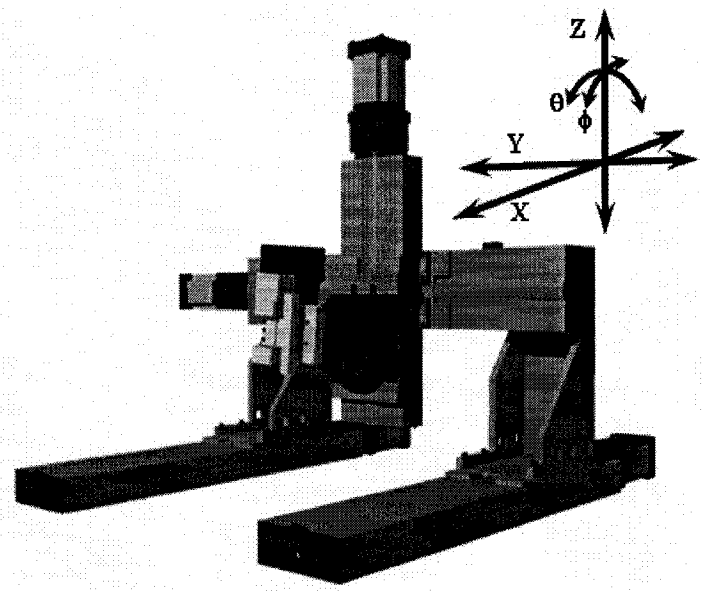


Figure 3.9: Schematic of 5-axis motion control system

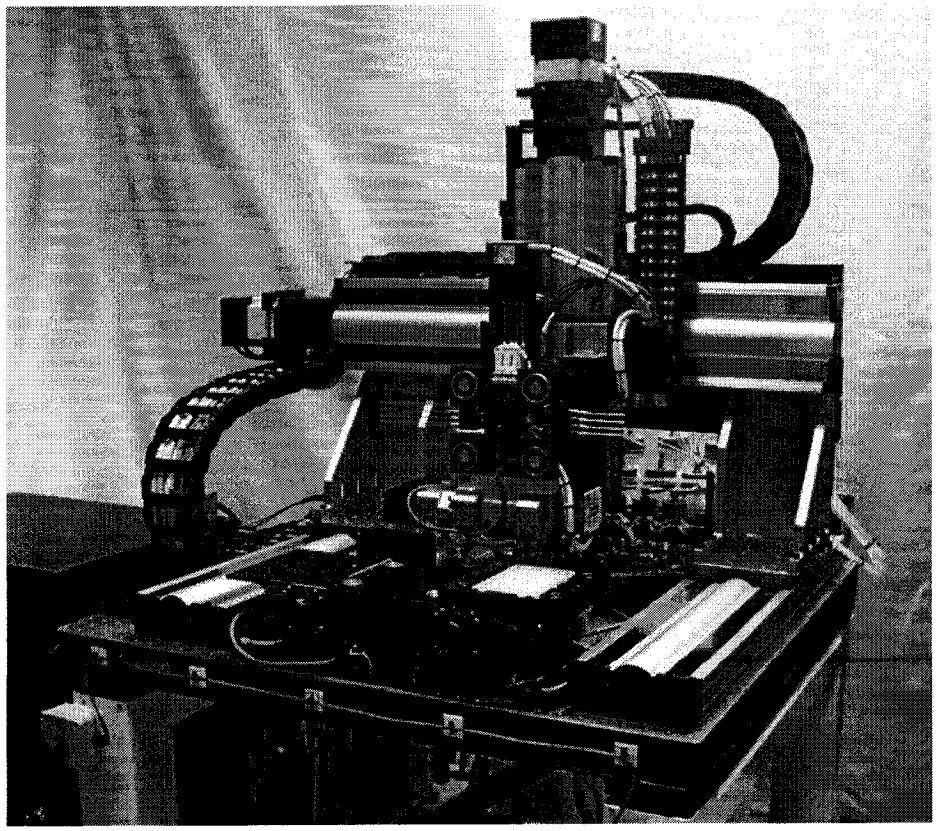


Figure 3.10: Photograph of 5-axis motion control system

Electromagnetic arm

Figures 3.11 and 3.12 are a CAD schematic and photograph of the electromagnetic arm, respectively. The arm is located on the ϕ-axis of the motion control system and acts to reconfigure the core mechanism during processing by picking up and decoupling the plates. Magnetic sensors located on the arm's two faces sense alignment between the arm and the plates. Once proper alignment is achieved, four H-bridge electromagnets, embedded on each face of the arm, attach to the steel discs of either the FTP, VPP or retainer plates. Decoupling occurs when the arm's electromagnets receive a quick polarity reversal pulse, which causes them to disengage from the corresponding plates' electromagnets and release the plates.

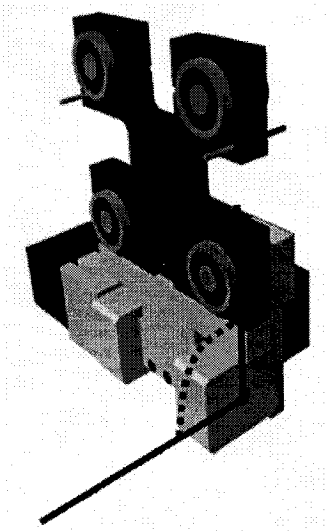


Figure 3.11: Schematic of electromagnetic arm

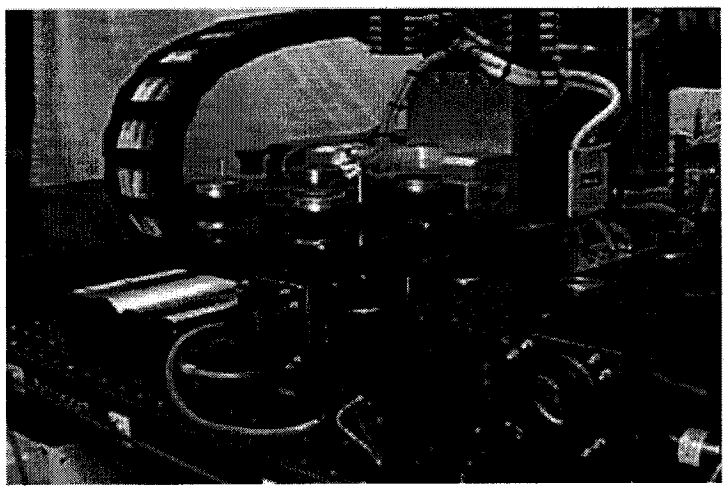


Figure 3.12: Photograph of arm after decoupling the FTP

Transfer platform

The transfer platform unit serves as temporary placeholder for the inverted core mechanism when the plates are being transformed from the processing configuration, to the delivery configuration. The transfer platform includes a platform, which can be lowered (Figs. 3.13A and 3.14A) or raised (Figs. 3.13B and 3.14B) to accommodate the inverted configuration. The platform is shaped such that the arm has directed passage through it.

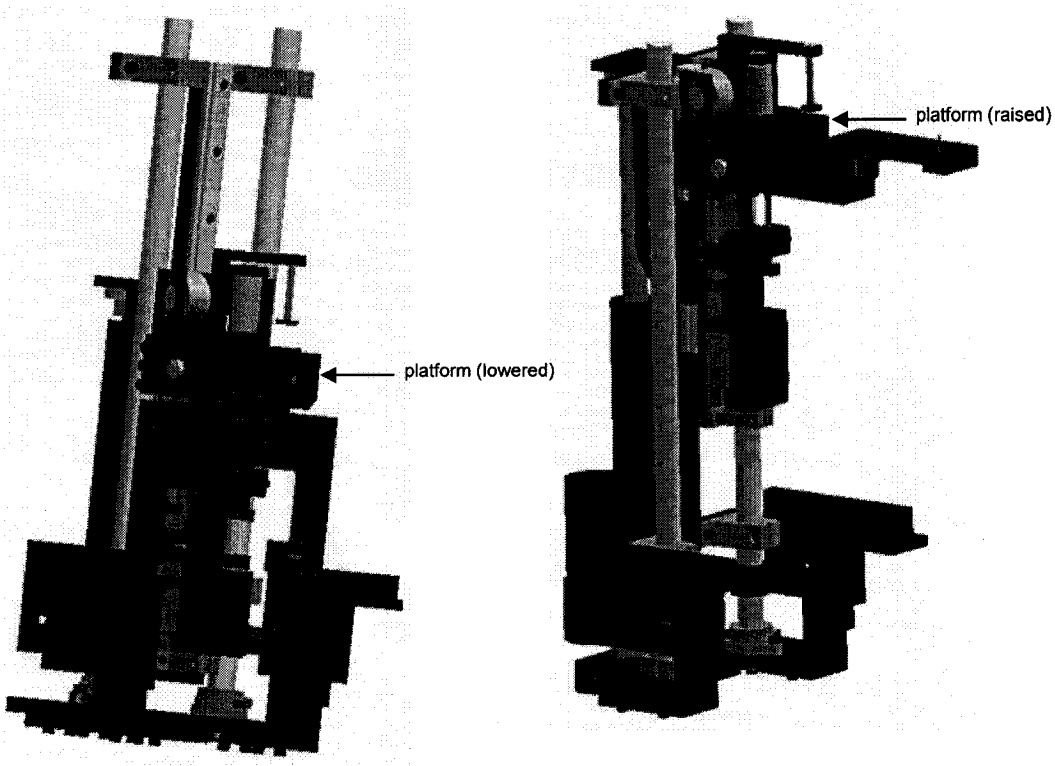


Figure 3.13: Schematic of transfer platform: (A) lowered platform, (B) raised platform

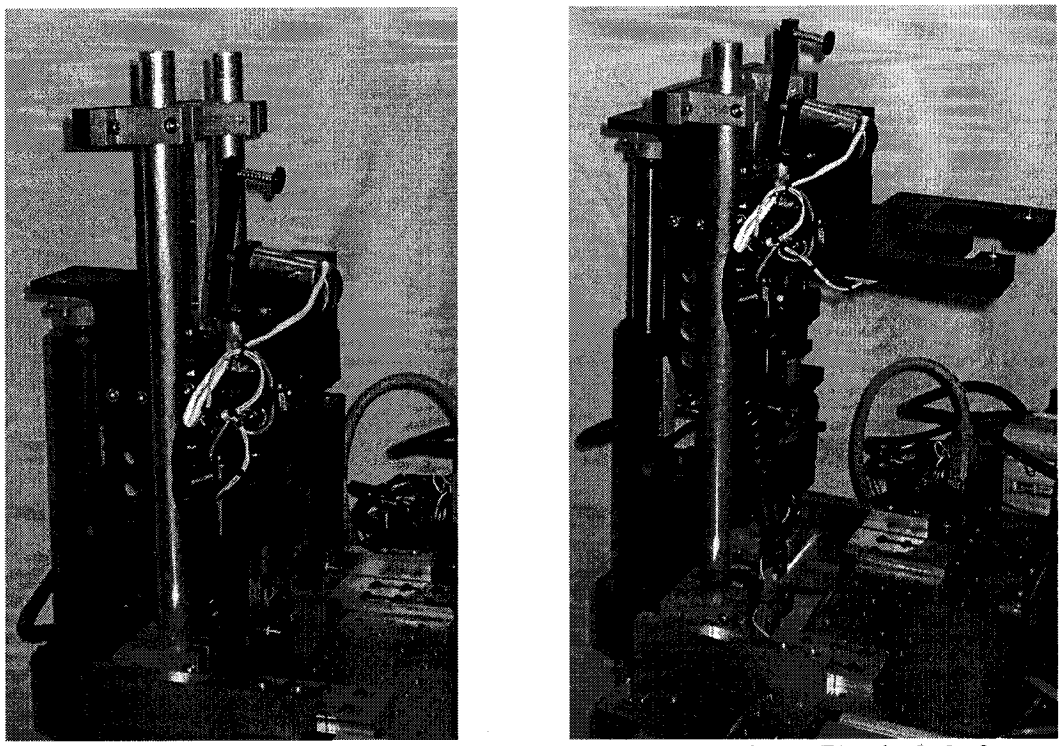


Figure 3.14: Photograph of transfer platform: (A) lowered platform, (B) raised platform

The sequence of events that occurs to reassemble the plates to form the delivery configuration are as follows:

(1) The arm lifts the processing configuration (Fig. 3.15)

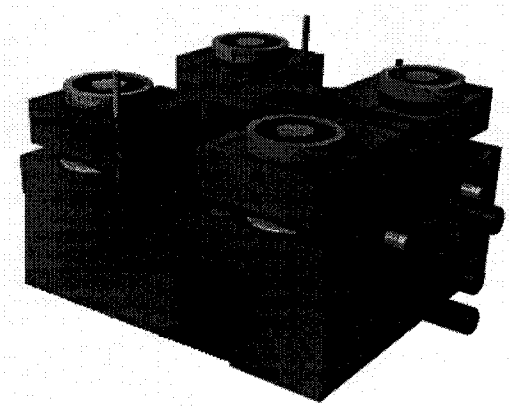


Figure 3.15: Arm lifts processing configuration

(2) The arm inverts the processing configuration by 180° and places it on the platform's raised surface (Fig. 3.16)

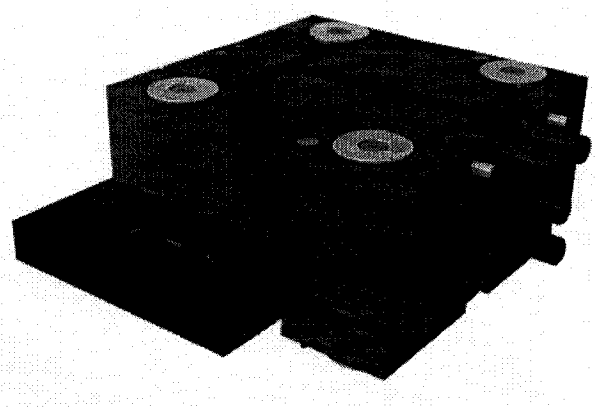


Figure 3.16: Arm inverts processing configuration and plates it on the transfer platform

(3) The arm which is now located underneath the plates, detaches itself from the assembly and passes through the platform's surface (Fig. 3.17)

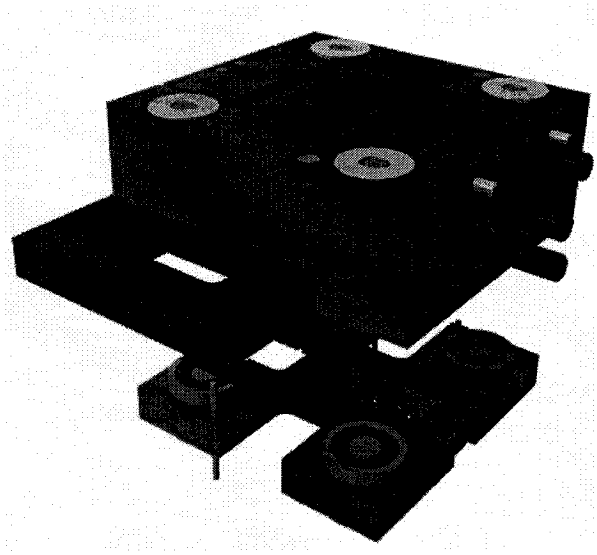


Figure 3.17: Arm detaches from inverted configuration and passes through transfer platform's raised surface

(4) The arm repositions itself above the inverted assembly and picks up the newly formed delivery configuration (Fig. 3.18)

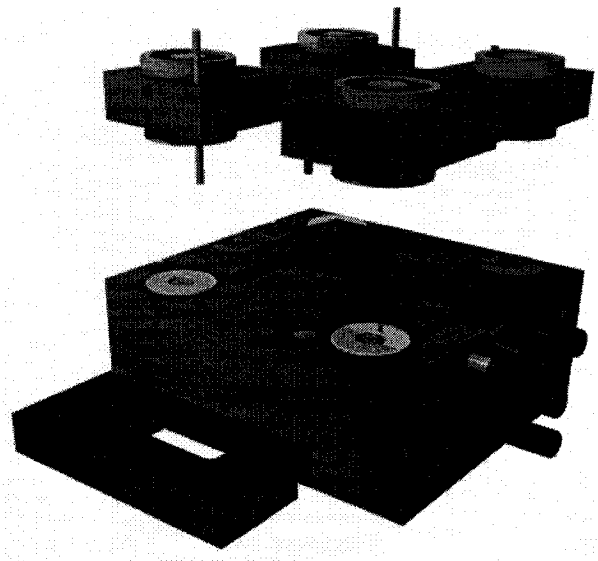


Figure 3.18: Arm repositions itself above the delivery configuration

Pneumatics manifold

The primary function of the pneumatics manifold (Fig. 3.19) is to supply the robot's VPP with vacuum and pressure for sample filtration and ejection. A compressor coupled directly to two pneumatics reservoirs, one for vacuum and pressure, generates the required amounts of vacuum and pressure. Delivery of either air source to the VPP (Fig. 3.20) is achieved by actuating a series of direct acting and proportional solenoid valves. Two sets of transducers, attached to the cylinders and the pneumatic lines, measure the pressure within the reservoirs, and experienced by the samples.

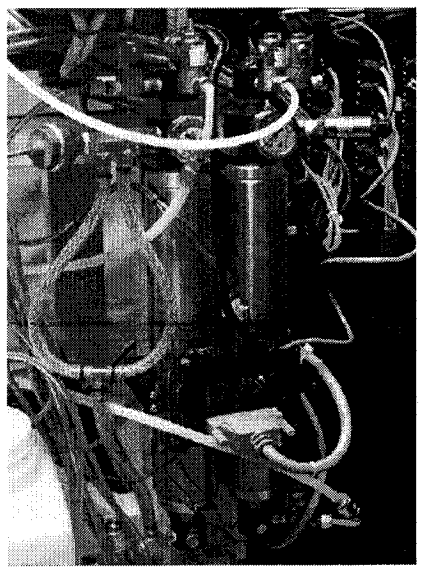


Figure 3.19: Photograph of pneumatic system

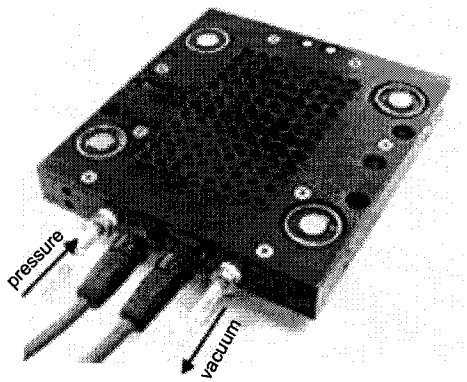


Figure 3.20: Photograph illustrating the delivery of pressure and vacuum to the VPP

Fluids handling mechanism

The fluids handling subsystem incorporates mechanisms to: (1) deliver reagents, (2) aspirate reagents, (3) clean needles and (4) remove system waste.

Deliver reagents

This mechanism is responsible for priming the system's fluid lines and delivering a specified volume of reagent to the wells of the core mechanism; it is located on the CP base. The chemical reagents necessary for sample preparation are stored in 19 bottles located at the rear of the robot. Figure 3.21 shows the robot's fluids handling panel. The panel is mounted to the back of the Y-axis, consisting of 2- and 3-way directional solenoid valves, a mixer, and two stepper pumps, which guides the reagents to a needle manifold located on the robot's θ-axis. The needle manifold, as seen in the photograph (Fig. 3.22), comprises three sets of needles, for: (1) processing, (2) embedding and (3) aspirating. The processing needles are used to dispense the majority of the reagents for the sample preparation protocol, whereas the embedding needles are used to dispense reagents that are specific to the Embedding stage such as Propylene oxide and Epon. The reason for having two sets of delivery needles will be made clear in the next chapter.

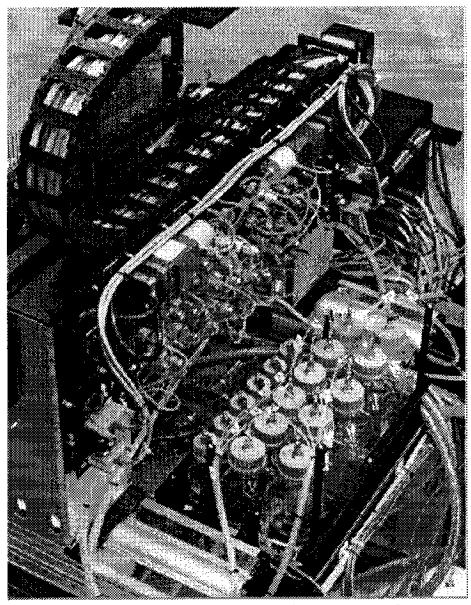


Figure 3.21: Photograph of fluids handling panel

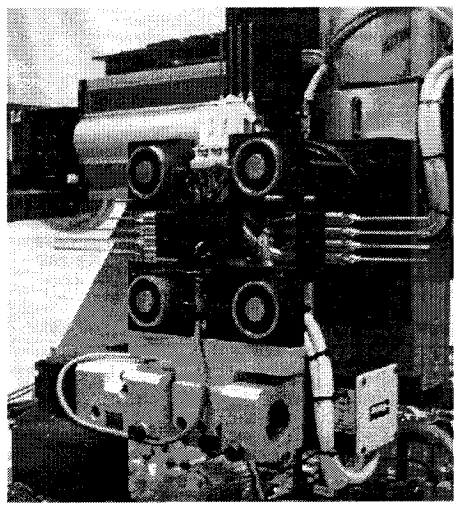


Figure 3.22: Photogram of needle manifolds

Figure 3.23 shows the organization of the fluids handling panel. Each reagent line is attached to a single valve (depicted by the small circle) on manifolds 1, 2, or 3. Manifolds 1 and 2 direct reagents through pump 1, via valves A1 to B1, while the third manifold directs reagents through pump 2 via valves A2 to B2. Valves C1 and C2 are 3-way valves and steer reagents either to valve D1 and D2 or to the valves on manifold 4. Valves D1 and D2 lead the reagents coming from pumps 1 and 2 into the mixer. Manifold 4 serves to direct fluids coming from the mixer or the two stepper pumps to either the processing or embedding dispense needles.

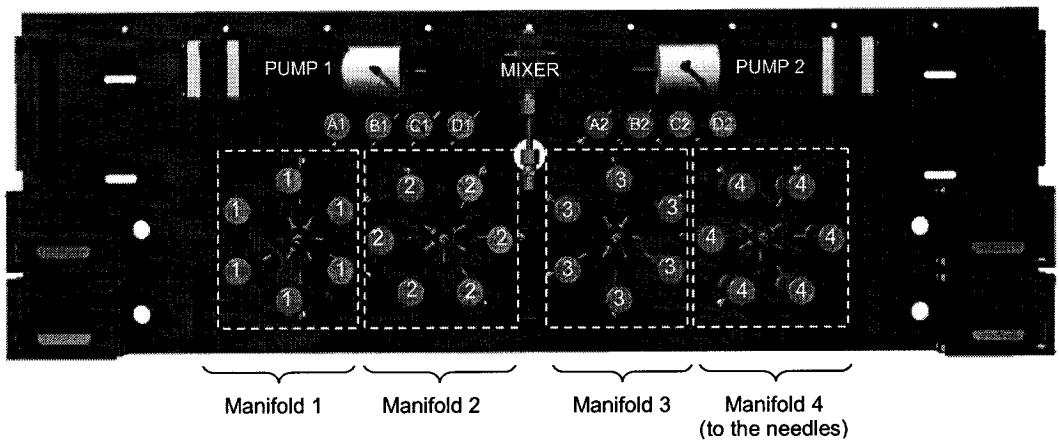


Figure 3.23: Diagram of fluids handling panel

Aspirate reagent

Before dispensing a fresh reagent to the samples, the unwanted reagent must first be aspirated from the core mechanism. Aspiration occurs when a valve attached to the vacuum cylinder is actuated to apply a small amount of negative pressure.

Clean needles

Needle cleaning takes place at the robot's clean station (Fig. 3.24). The clean station comprises of two compartments: (1) to wash needles, and (2) to dry them. Washing occurs by immersing the needles into the wash compartment, where fresh water is circulated. Conversely, drying is achieved by establishing a connection between the dry compartment and the vacuum reservoir.

Figure 3.24: Photograph of clean station

Remove system waste

The pneumatics system's final function is to expel waste from the robot. As the reagent lines are primed prior to dispensing and aspirated via the aspiration needle, unwanted reagent accumulates in the vacuum's reservoir. System waste is removed by actuating a sequence of valves and a gear pump to flush the vacuum cylinder.

3.3 Robot Operation

Now that we have examined the mechanical components of the EMPOP robot, we will establish how these six modular systems interact throughout a processing run. The 20 illustrations staggered below (Figs. 3.25 A-T) represent each stage of the automated protocol for a single well of the core mechanism.

Steps 1 and 2

Before the automated protocol can begin, the operator must first assemble the plates in the processing configuration and then pipette the proteomic organellar sample, which is suspended in a fixative solution consisting of Glutaraldehyde, Cacodylate buffer and Calcium chloride, into one or more of the 96 core mechanism wells (Figs. 3.25 A-B).

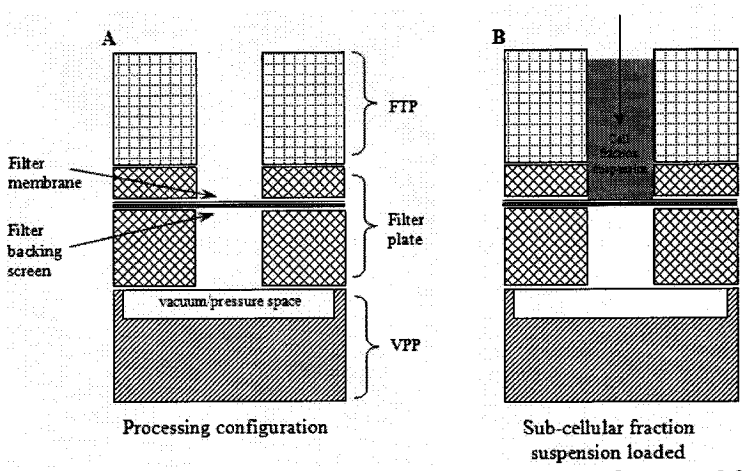


Figure 3.25A & B: Cross-sectional view of steps 1 and 2 of automated protocol for a single well

Step 3

Automation begins by vacuum filtering the samples onto the filter membrane via the VPP (Fig. 3.25 C).

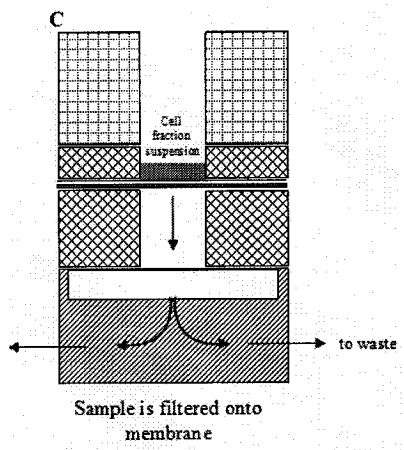
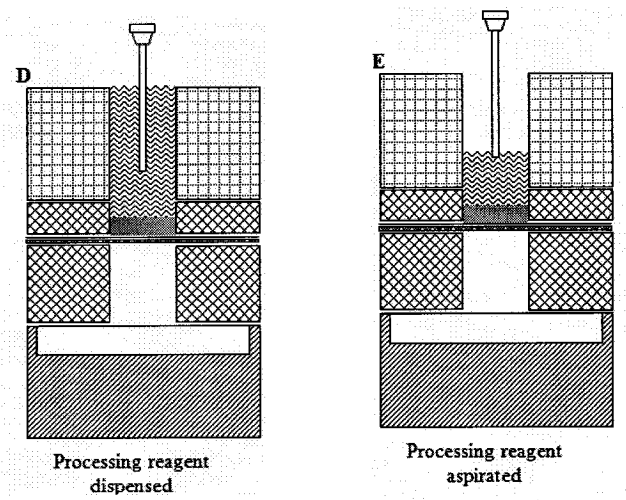


Figure 3.25C: Cross-sectional view of step 3 of automated protocol for a single well

Steps 4 and 5

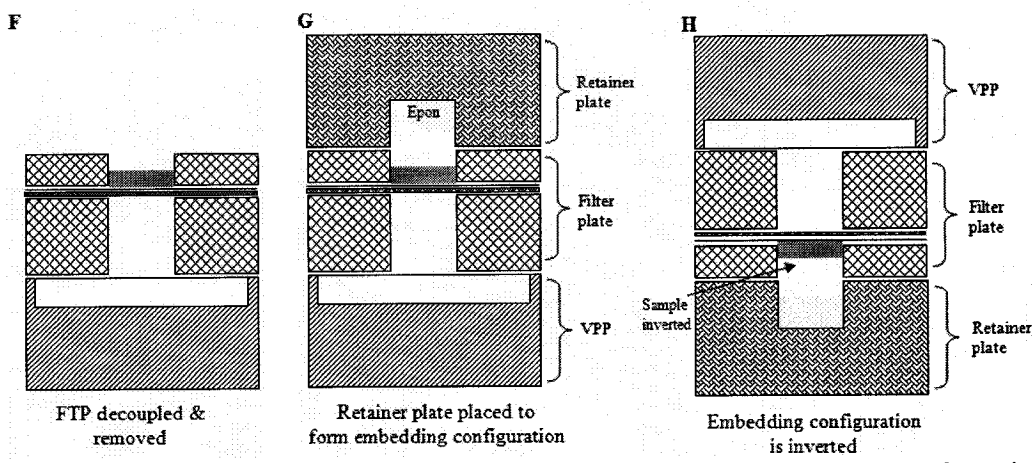
Following filtration the samples undergo a rigorous chemical treatment involving several iterations of reagent dispensing, incubating and aspirating (Figs. 3.25 D-E).



Figures 3.25D & E: Cross-sectional view of steps 4 and 5 of automated protocol for a single well

Steps 6, 7 and 8

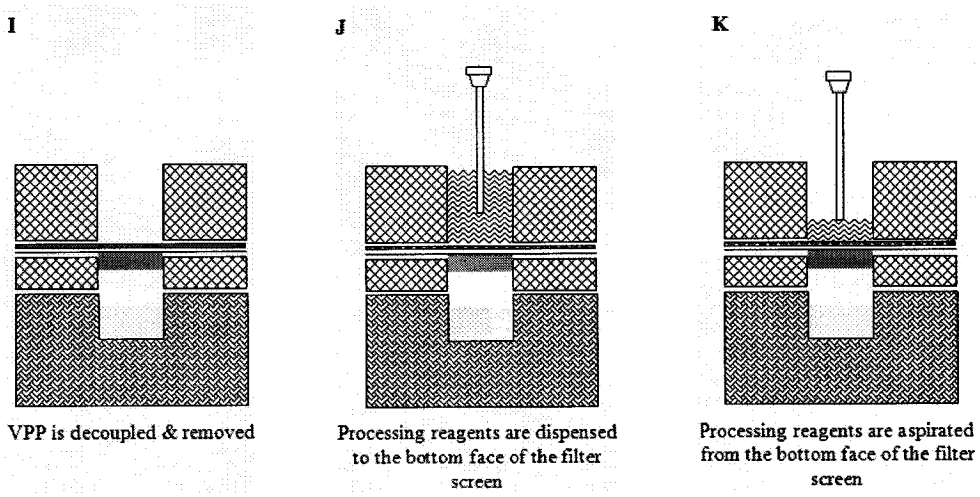
Prior to the last dehydration step, the electromagnetic arm rearranges the plates into the delivery configuration. This is achieved by having the electromagnetic arm decouple the FTP plate, replace the FTP plate with the retainer plate, and invert the core mechanism via the transfer platform (Figs. 3.25 F-H).



Figures 3.25F, G & H: Cross-sectional view of steps 6, 7 and 8 of automated protocol for a single well

Steps 9, 10 and 11

After inversion, the arm temporarily removes the VPP, revealing the embedding configuration. From the illustration, it is evident that the sample in both the delivery and embedding configurations is located beneath the filter screen. Consequently, chemical processing continues from the backside of the filter screen to permeate and dissolve the filter membrane (Figs. 3.25 I-K).



Figures 3.25I, J & K: Cross-sectional view of steps 9, 10 and 11 of automated protocol for a single well

Step 12

Chemical processing comes to a halt once the robot delivers the first pass of propylene oxide to the core mechanism. At this point, the robot reassembles the delivery

configuration and forces the samples onto the retainer plate by applying a pulse of pressurized air (Fig. 3.25 L).

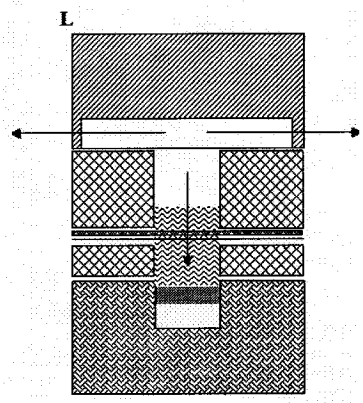
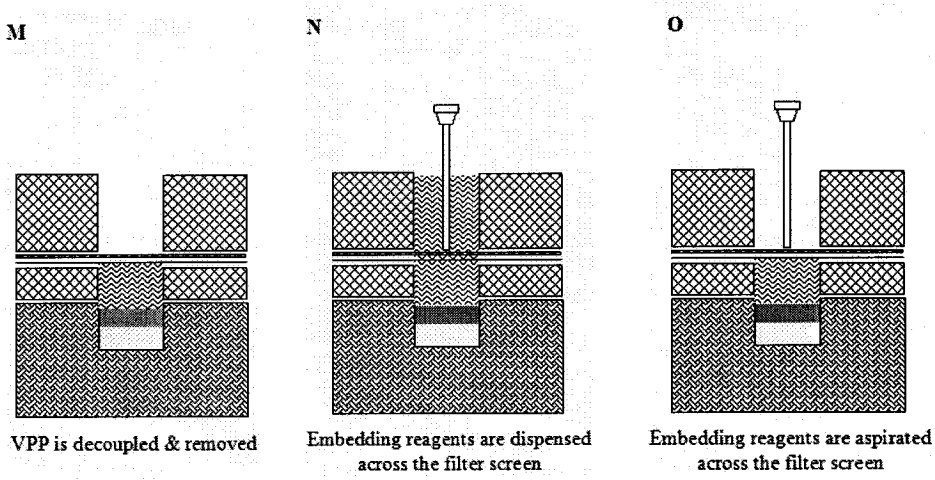


Figure 3.25L: Cross-sectional view of step 12 of automated protocol for a single well

Steps 13, 14 and 15

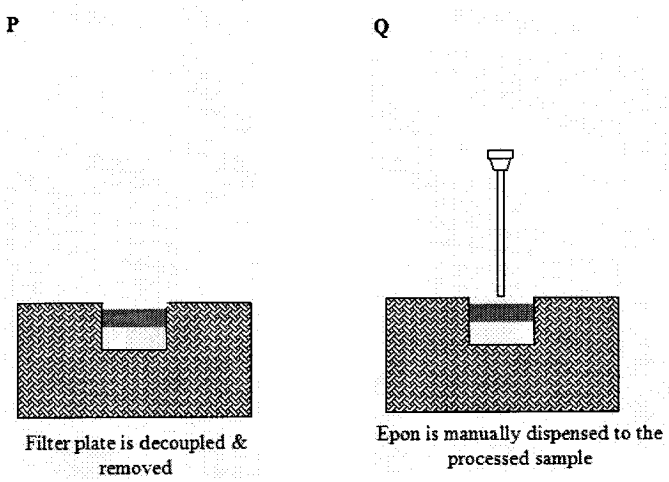
Subsequent stages require the arm to remove the retainer plate (forming the embedding configuration) to prepare the samples for the manual application of polymerizing resin (Figs. 3.25 M-O).



Figures 3.25M, N & O: Cross-sectional view of steps 13, 14 and 15 of automated protocol for a single well

Steps 16 and 17

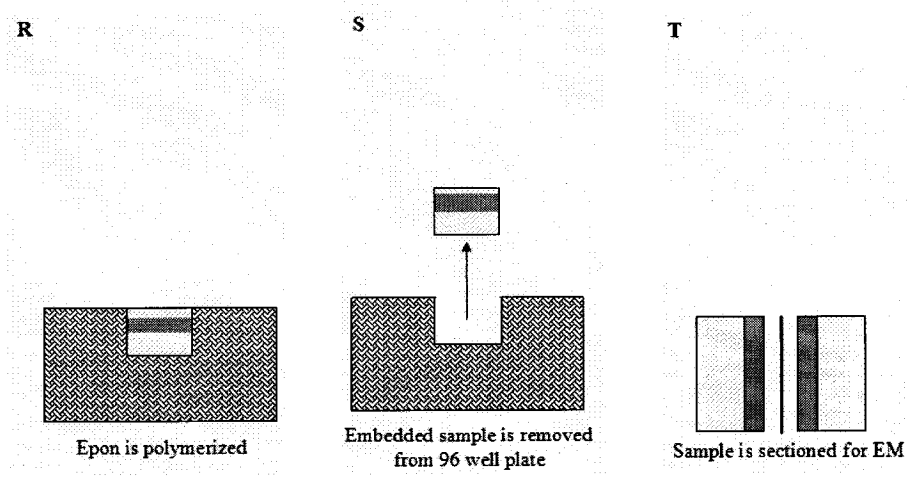
The last stage of automated processing involves decoupling the filter plate (Fig. 3.25 P); after which, the operator must continue processing by applying a final layer of Epon to the 96 well plate (Fig. 3.25 Q).



Figures 3.25P & Q: Cross-sectional view of steps 16 and 17 of automated protocol for a single well

Steps 18, 19 and 20

When the samples are fully polymerized, they are manually removed from the 96 well plate and are presented to the microtome for sectioning prior to EM analysis (Figs. 3.25 R-T).



Figures 3.25R, S & T: Cross-sectional view of steps 18, 19 and 20 of automated protocol for a single well

4 System Architecture and Software Implementation

4.1 Chapter Overview

This chapter describes the system architecture and software developed to control the EMPOP robot. The chapter begins with an overview of the hardware and software methodologies employed. It follows with an in-depth examination of the robot's HMI organization and function, with special attention to the operational parameters that can be set by the operator. The chapter concludes by describing the algorithms developed to control the five main EMPOP processes.

4.2 System Architecture

Industrial robots serve many different purposes in a wide variety of industries, ranging from garment fabrication to car manufacturing. In general, they perform repetitive, laborious, time-consuming, and dangerous tasks otherwise completed by humans. Apart from these advantages, the key to their increasing success is that they permit individuals with limited technical knowledge of robotics or automation to operate them easily and produce repeatable, consistent results.

Initially, roboticists designed robots to mimic human function while eliminating human error. The three main systems associated with human function are a muscular/skeletal system, a sensory system and a nervous system. Within the realm of robotics, these equate to electrical/mechanical hardware components, feedback sensors, and a control module. In the human body, the nervous system drives the musculo/skeletal system via sensory information it receives. Similarly, automated sample preparation requires a control module to control the robot's electrical/mechanical systems using feedback from sensors.

To perform its task of controlling the EMPOP robot, the control unit was programmed to 'think and react'. In addition, and of utmost importance to the end-user, the control unit was instructed to obey and carry out orders made by the operator via an intuitive, high-

level, custom-designed, software application. This last criterion, from the standpoint of the buyer and/or the end-user characterizes a functional system[35,36].

The EMPOP robot uses a PC-based automation scheme based on a client-server software architecture. This scheme consists of two logical hardware units: (1) a PC or host computer and (2) a robotic unit consisting of a control module, sensors, and a combination of electrical and mechanical devices. Communication between the PC and the robot is achieved using the TCP/IP protocol across an Ethernet link. Once a connection is established, the host computer communicates information to, and receives status updates, from the robot. Information is transmitted between the hardware units by software using the client-server methodology, where the client makes a request to the server, which on reception fulfills the client's request.

The benefit of the two-tier hardware/software architecture is that it creates two functional levels each with distinct responsibilities and features. Moreover, it prevents a novice operator from directly accessing and/or modifying the system's circuitry or embedded control algorithms, which could easily disrupt the robot's function. The goal of the upper/operator echelon (PC and client software) is to create a familiar, friendly environment where the operator can interact easily and efficiently with the robot, without having to understand the system's intricate electrical and mechanical structure and function. Once the operator sets the operational parameters, the HMI application sends a single high-level command to the EMPOP robot, which initiates processing. Thereafter, the low/control echelon (robot and server software) assumes responsibility for all aspects of robotic function. Consequently, the control unit must be capable of fast, real-time processing since timing and coordination of the robot's subsystems are crucial for proper system behavior.

Figure 4.1, shows the general structure of the two-tier system. The EMPOP system's operator tier uses a standard PC running Windows XP as its host computer. The client software running on the host computer consists of a HMI application developed as part of this thesis and written in Microsoft Visual Basic™ (VB).

The control unit is a Compumotor 6K8 motion controller[37]. This a hybrid controller that combines, within a single package, all the circuitry necessary for motion control as well as the I/O needed for controlling motor and drive function, monitoring sensors, and actuating peripheral devices. Many different control elements were required to enable the robot to perform its various tasks; consequently, the standard system was expanded with 12 additional I/O cards, resulting in a total of 96 I/O channels – 48 digital outputs, 24 digital inputs, 16 analog outputs, and 8 analog inputs[37].

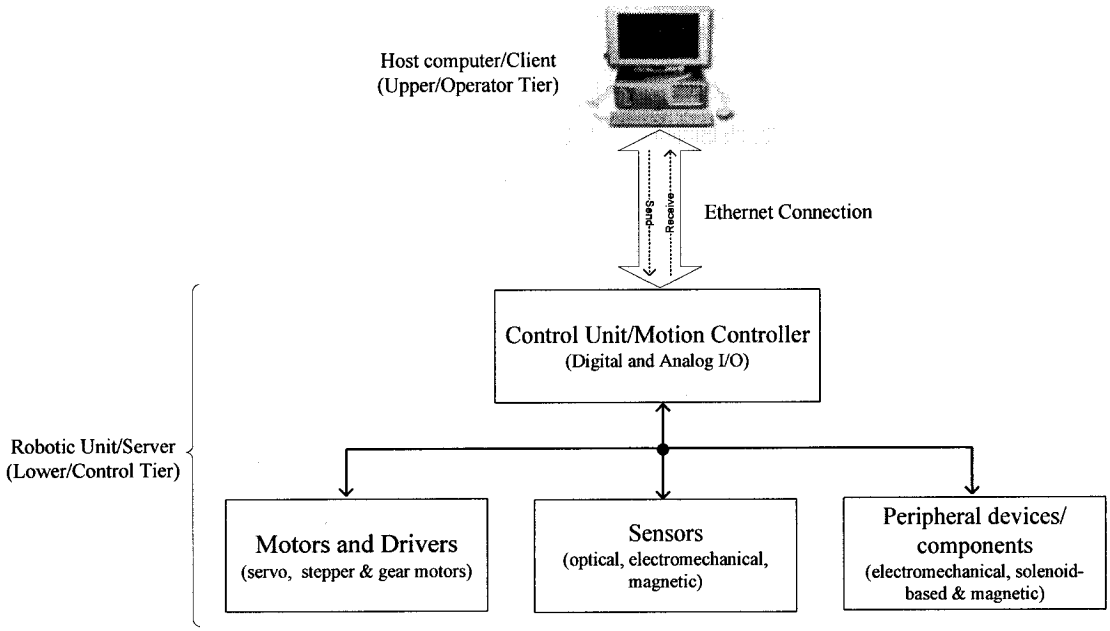


Figure 4.1: Logical diagram of two-tier system

The control elements that comprise the EMPOP robot consist of: (1) motors and drivers, (2) sensors and (3) peripheral devices. These are controlled and coordinated via a library of embedded process control routines developed using MotionPlanner™ (MP), a language specific to the Compumotor controller[37]. These routines were written to work together to orchestrate all aspects of the automated system.

4.3 Operator Tier

The operator layer of our two-tier system is the only method for the operator to interact with the EMPOP robot. The goal behind developing this layer was to create a familiar

environment for individuals with minimal computer expertise. This was accomplished by developing a powerful and flexible human-machine interface (HMI) application, to accommodate the operator's needs. In addition, it had to be capable of organizing data received from the control unit; to enable the operator to monitor system function during a run, and validate it against previous runs.

VB was chosen to write the HMI application as it has a reputation of having shorter program development times, than other languages such as Microsoft Visual C++[38]. Shorter development times are due largely to the VB programming environment, which allows programmers to easily create and design complex GUIs using predefined, customizable objects, thus reducing the need for API and system kernel calls. As a result, VB is the preferred programming language when developing visual applications that do not stray far from the standard windows 'look and feel'. Its major drawback is that it does not support real-time control. However, in our case, the use of a two-tier system architecture overcomes this problem, since the control-tier, via the control unit, controls all processes and communicates information to the host-computer in real-time; this eliminates the need for real-time processing at the operator-tier.

The robot's HMI consists of six graphical user interfaces (GUIs): (1) Ethernet connection, (2) Calibration Setup, (3) System Setup, (4) Reagent Setup, (5) Process Control and (6) Process Monitoring. The following sections will discuss each GUI, with emphasis on the parameters that can be modified by the operator.

Ethernet Connection

This GUI, shown in Figure 4.2, enables the operator to connect to the EMPOP system using its IP address. It also provides three dialog boxes, which permit the operator to upload a configuration file and/or save data in a log file.

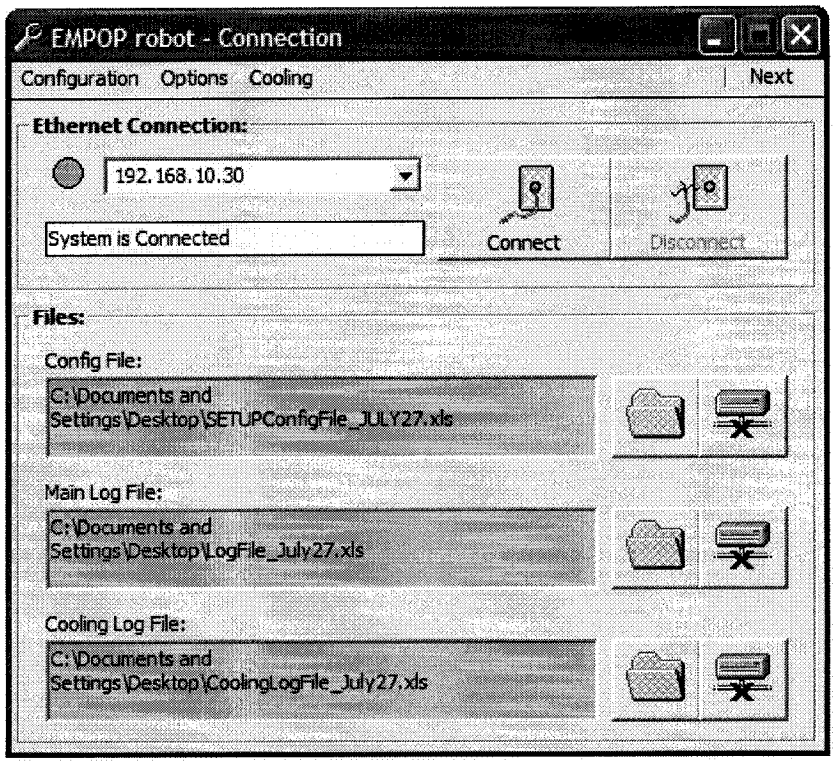


Figure 4.2: Ethernet Connection GUI

The configuration file is an Excel spreadsheet that contains all user-defined parameters specified by the GUIs. Loading this file makes it possible to repeat a processing run with a fixed set of parameters. Moreover, if the operator makes changes to any parameters during a run, the existing configuration file may be updated or a new configuration file created.

Two log files are used to store robotic data. The main log file is used to save robotic process and sensory information during processing. Since information is appended within the log file as it arrives, it is crucial that the log file be well organized. To maintain the clarity and readability of the main log file, a second log file, the cooling log file, is used to capture temperature data, which is received by the HMI application every 2.5 seconds. This avoids cluttering the main log file with multiple temperature readings.

Calibration Setup

At start up, the motion controller does not know the location of the system's plates and CP relative to the dispense and aspiration needles, and the electromagnetic arm. As a

result, the system does not know where to dispense or aspirate reagents, or where to retrieve or decouple the plates. Consequently, before processing can begin, the operator must specify the X, Y, Z, θ and ϕ axes positions for the plates and CP, starting from a fixed reference location. Needle calibration involves determining the location of each of the three needle manifolds with respect to the first and last channels of the two- and/or three- plate configuration, and with respect to the clean station. Electromagnetic arm calibration requires the operator to resolve the system's X, Y, Z and θ positions, and in addition determine the arm's ϕ extension angle needed to: (1) drop and pick-up plates, to or from, the CP's two surfaces; (2) decouple the top plate from the dispensing or delivery configuration; and (3) temporarily place and retrieve the inverted dispensing configuration onto or from the transfer platform. Table 4.1 lists the calibration locations for the needles and arm. These locations can be calibrated in any sequence. However, when calibrating the needles it is more efficient to calibrate a single location for all needle types before proceeding to the next location.

The GUI (Fig. 4.3) supports two calibration methods. The first method enables the operator to move a single axis by either a predefined incremental amount or any absolute distance. In incremental displacement, each movement is made in reference to the system's current position, whereas in absolute mode, distances are specified with reference to a home or zero location. Incremental positioning is generally used to make small adjustments to the current position, whereas absolute positioning is used to quickly move the axis to an area close to its final destination.

The second method differs from the first, in that it simultaneously moves all axes to a previously defined calibrated location; the operator need only fine-tune each axis's coordinates to finalize calibration. Although calibration using the second method greatly reduces the time and effort needed to calibrate the system, it does require the operator to have either uploaded a past configuration file or modified the current configuration file to include past calibrated valves.

CALIBRATION	LOCATIONS
Dispense Needle	Three plate configuration – Channel A1
Dispense Needle	Three plate configuration – Channel H12
Dispense Needle	Two plate configuration – Channel A1
Dispense Needle	Two plate configuration – Channel H12
Dispense Needle	Wash Station
Dispense Needle	Dry Station
Aspiration Needle	Three plate configuration – Channel A1
Aspiration Needle	Three plate configuration – Channel H12
Aspiration Needle	Two plate configuration – Channel A1
Aspiration Needle	Two plate configuration – Channel H12
Aspiration Needle	Wash Station
Aspiration Needle	Dry Station
Embedding Needle	Two plate configuration – Channel A1
Embedding Needle	Two plate configuration – Channel H12
Embedding Needle	Wash Station
Embedding Needle	Dry Station
Electromagnetic Arm	Cooling Platform base
Electromagnetic Arm	Cooling Platform A
Electromagnetic Arm	Cooling Platform B
Electromagnetic Arm	Transfer Platform

Table 4.1: Needle and arm calibration locations

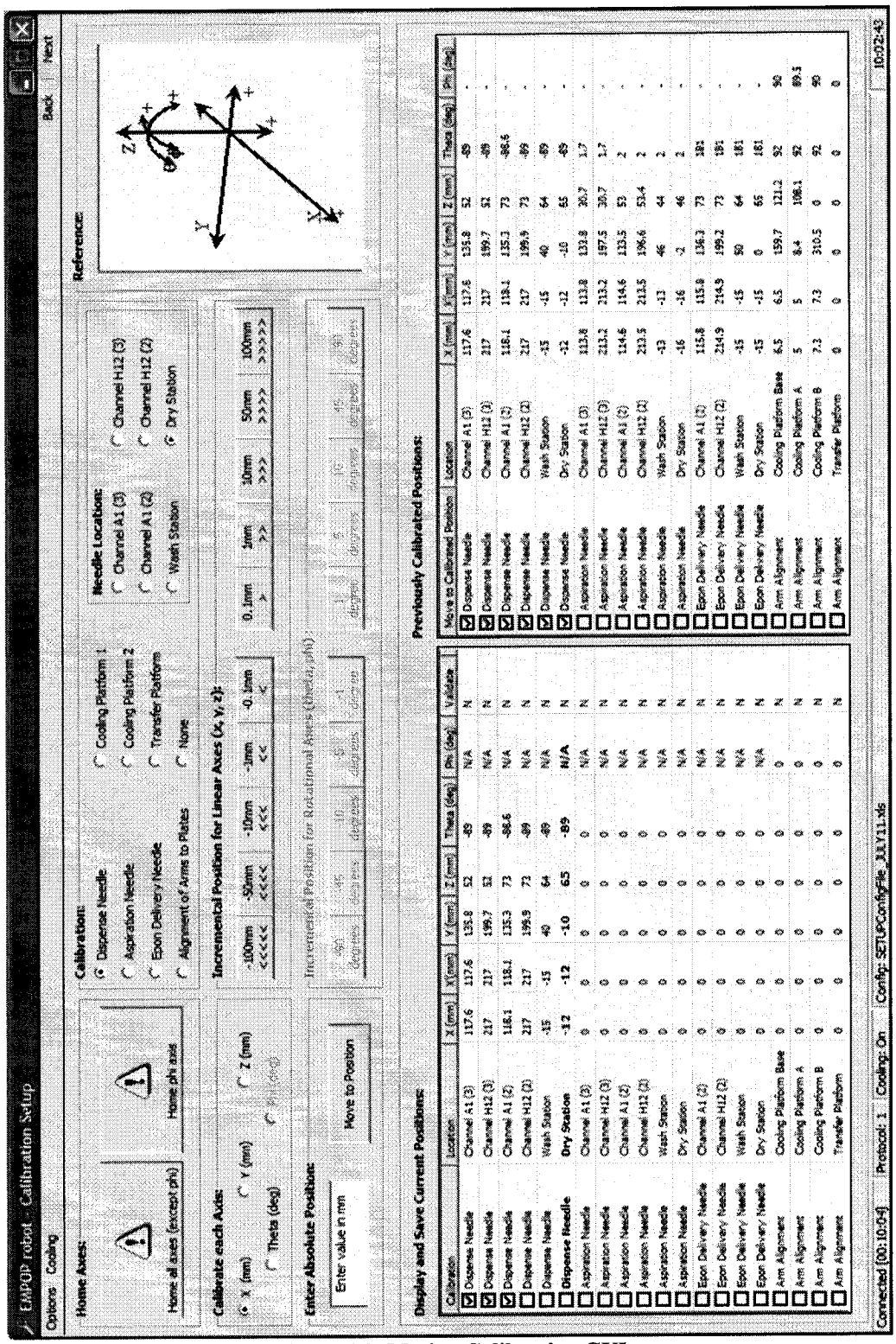


Figure 4.3: Motion Calibration GUI

The Calibration Setup GUI displays the absolute position for all axes after each move. This enables the operator to view the coordinates for all the calibrated locations. Once the operator is satisfied with the X, Y, Z, θ and ϕ axes positions for a particular location, the HMI downloads the coordinates into the system's motion controller, where they are stored in a pointer data structure. This procedure continues until all the locations are calibrated. Thereafter, since needle calibration only establishes the positions of the first and last wells, the controller must interpolate the locations of the 94 remaining wells. These newly calculated positions are then stored in a separate data structure, where low-level routines can retrieve them as needed.

System Setup

For clarity and ease of use, the System Setup GUI, shown in Figure 4.4, is divided into three main sections, which deal with parameters for needle setup, needle washing, and well selection. Once set, the HMI downloads these parameters to the robot's microcontroller, where they remain fixed throughout a run.

Needle setup

There are two parameters associated with each needle type (i.e. processing, aspiration and embedding) on the needle manifold. The first parameter allows the operator to choose the number of needles to use (1, 2 or 4). The second enables them to set the first needle (A, B, C or D) to use for reagent dispensing or aspiration.

Needle washing

Needle washing is characterized by three parameters. The first two deal with 'how' the needles are cleaned and consist of: (1) the number of needle washes and dries, and (2) the duration of each wash. The third, the washing frequency, determines 'when' the needles are cleaned. This last parameter was implemented to enable the operator to minimize sample contamination between wells as a result of reagent dispensing or aspiration via tainted needles; it permits needles to be cleaned after each well is processed as opposed to waiting for all wells to be processed.

Well selection

This section allows the operator to downsize automated sample preparation, by controlling the number of samples that are prepared in sequence.

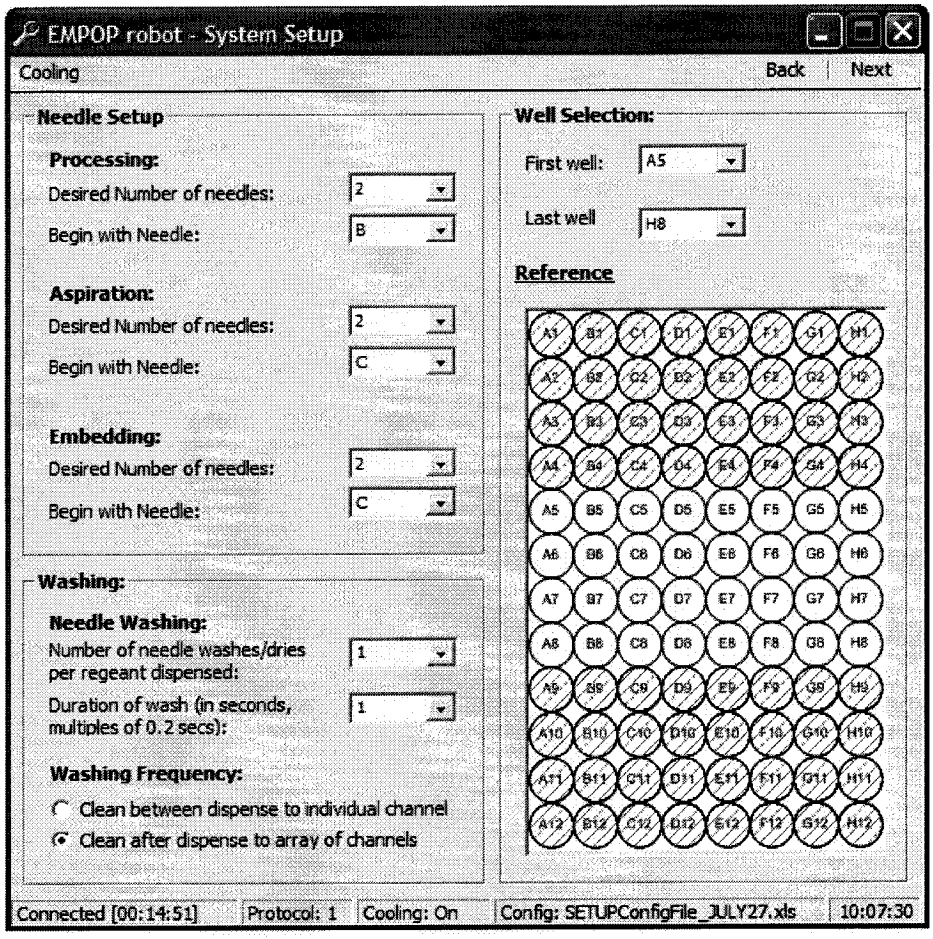


Figure 4.4: System Setup GUI

Reagent Setup

This GUI (Fig. 4.5) allows the operator to select a preparation protocol as well as specify what reagent each bottle contains. The EMPOP robot is equipped with 19 reagent reservoirs, of which 13 direct fluids through pump 1 and the remaining 6 through pump 2. The operator can place processing reagents in any bottle, providing they abide by two reagent reservoir rules, which are enforced by the GUI. The first rule states that reagents being mixed must be associated with different pumps. The second rule obliges all reagents containing propylene oxide to be contained in reservoirs that are associated

with the same pump. This last rule was put into effect to ensure consistent delivery of these reagents as they have a tendency to disassociate into droplets when pushed through air. Pump 2 was used for this since it has shorter tube lengths and more check valves, and thus minimizes the introduction of air into the fluid lines

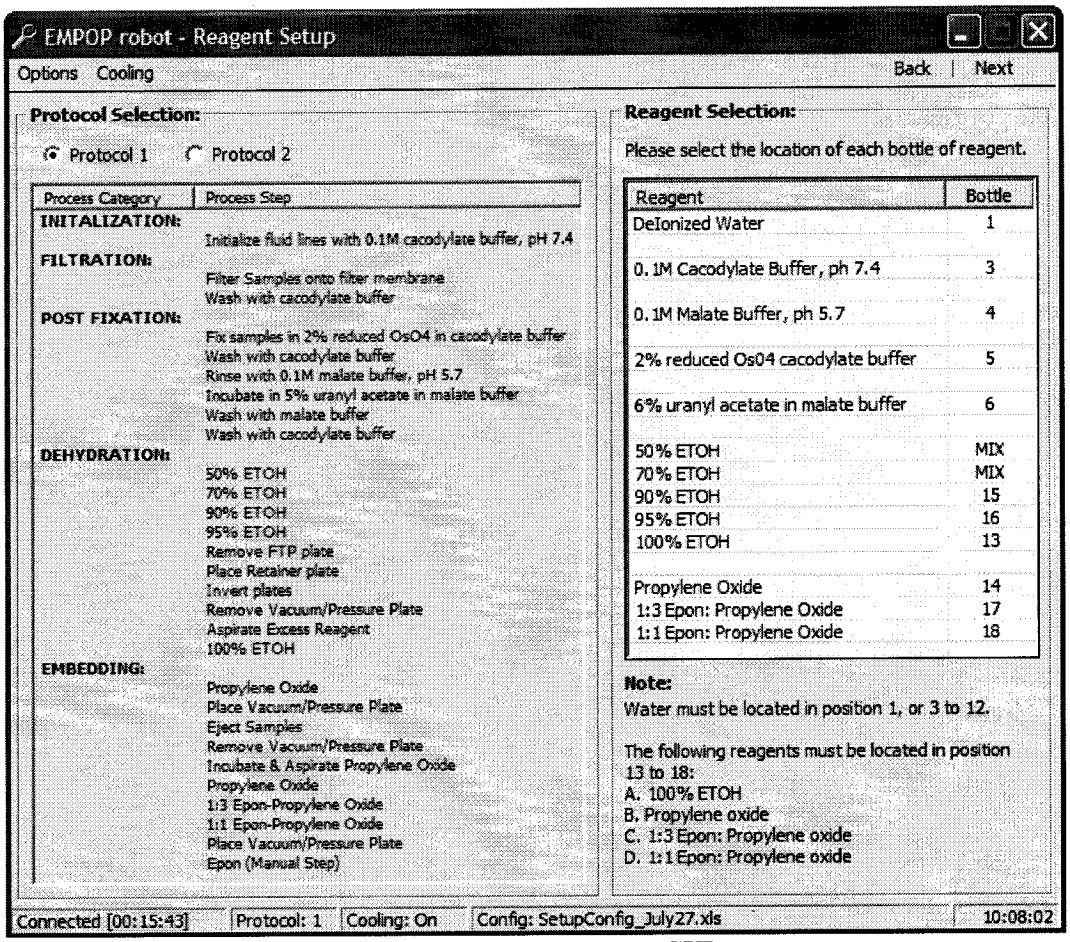


Figure 4.5: Reagent Setup GUI

Once all reagents are assigned to reservoirs, the HMI saves the information to a high-level data structure for use by the Process Monitoring GUI. This data structure includes information regarding:

(1) The link between each bottle and its corresponding valve on the fluid handling panel

(2) The type of delivery needle (processing or embedding) each reagent is dispensed from

(3) The plate configuration (2 or 3) each reagent is dispensed into

(4) The velocity with which each reagent is pumped

Process Control

This GUI (Fig. 4.6) allows the operator to select the mode in which the Process Monitoring GUI will run the robot – automatic, manual or combined.

In automatic mode, the robot executes each step of the protocol in sequence without user intervention. In manual mode, the operator must command the robot to start each step. This permits the operator to adjust the protocol during processing, by changing parameter values for filtration, reagent delivery/aspiration and ejection. Consequently, this feature gives the operator more control over the protocol; especially before the optimal parameter settings required for sample preparation are established. Moreover, in manual mode, the operator can execute steps in any sequence and as many times as they want. The disadvantage of operating in manual mode is that the operator must be present for the entire protocol, especially during the initial stages.

In combined mode the operator can customize operations by tagging some steps of the protocol to execute in automatic mode and others in manual mode. This allows the speed advantage of automatic mode to be combined with the flexibility of manual mode.

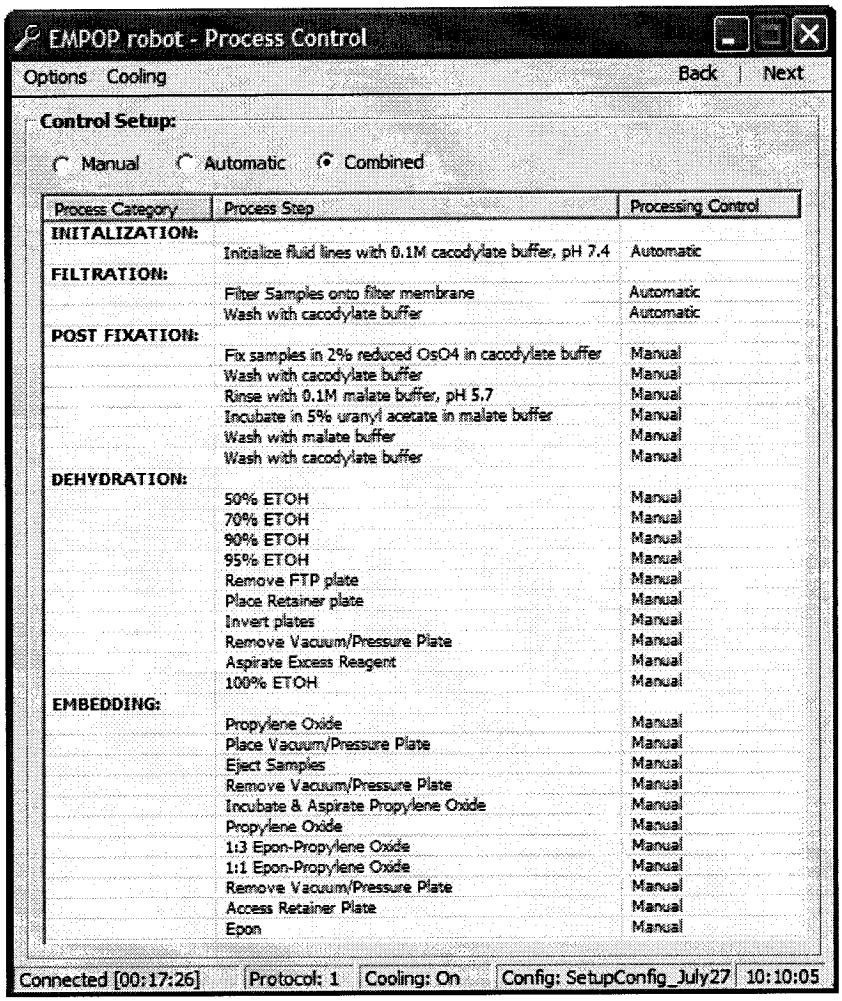


Figure 4.6: Process Control GUI

Process Monitoring

The Process Monitoring interface (Fig. 4.7) allows the operator to set and pass five operational parameters to the controller. It also calls the control unit's first-order layer of low-level, embedded control routines to execute sample filtration, fluid handing, plate manipulation and sample ejection.

This GUI consists of a tabular layout with nine columns. The first two columns indicate the processing category and step. The next five columns allow the operator to set parameters for each step; column 6 is used to start a processing step, and column 7 displays its status. A detailed description of columns 3 through 9 are listed below:

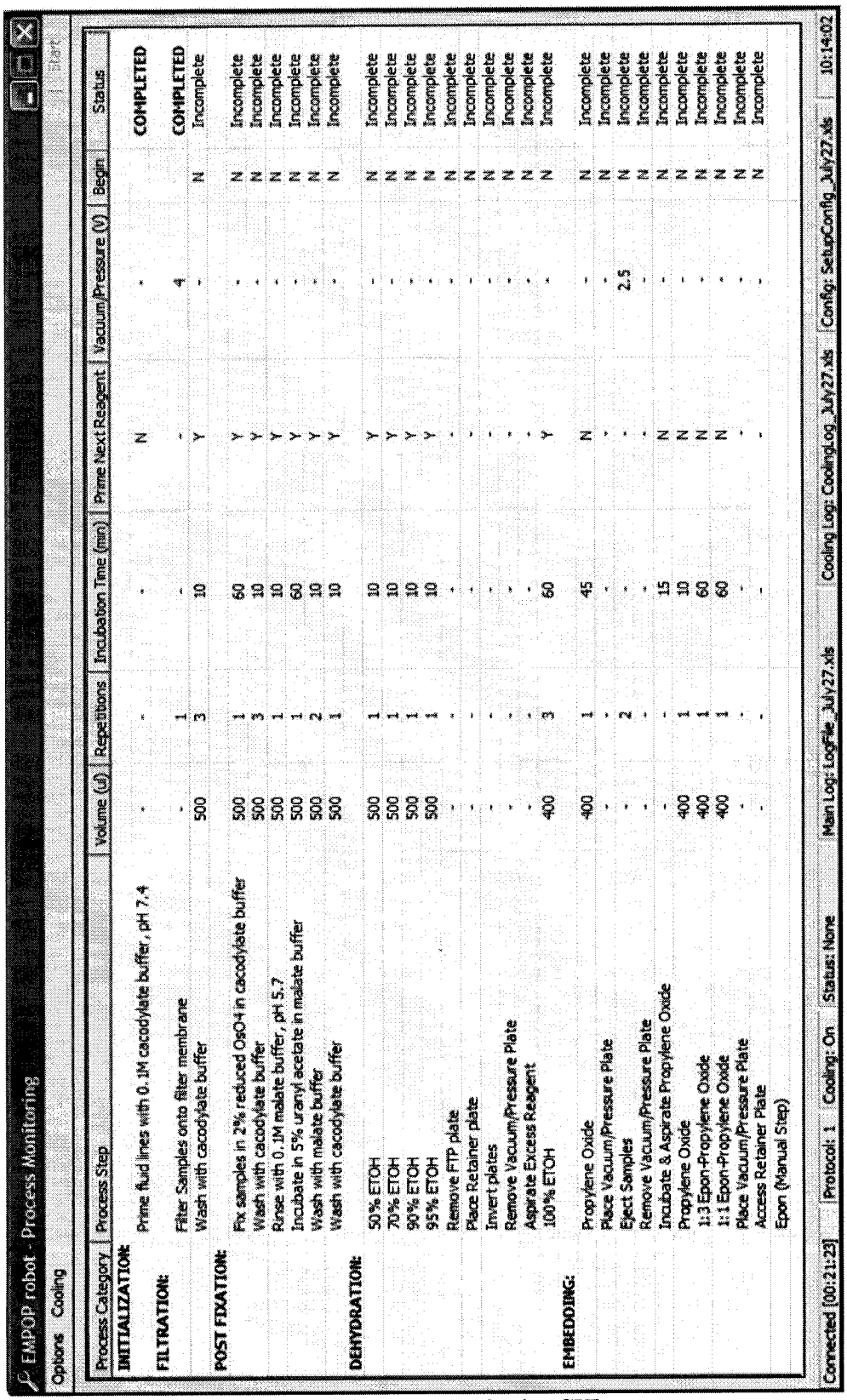


Figure 4.7: Process Monitoring GUI

(1) The `Volume` parameter specifies the volume of reagent, in µl, to dispense to each channel

(2) The `Repetitions` parameter defines the number of times a step is processed

(3) The `Incubation Time` parameter determines the duration, in minutes, for the samples to incubate

(4) The `Prime Next Reagent` parameter, is a yes/no flag that specifies whether the robot should prime the fluid lines with the next reagent during the current step's last repetition

(5) The `Vacuum/Pressure` parameter determines the volume to empty, or fill the system's two variable volume pneumatic reservoirs to generate the required amount of vacuum or pressure for filtration or ejection, respectively.

(6) The begin column, is a yes/no flag that the operator can use to initiate execution of a step in manual mode

(7) The status column indicates whether a step is complete or incomplete

Unlike the system setup parameters, the parameters set within this GUI are not immediately sent to the robot's motion controller. Instead, the HMI application transfers the information associated with each step just prior to calling the control routine responsible for it.

4.4 Control Tier

The control tier is the lowest level of the two-tier hardware/software system architecture. This layer is invisible to the operator so it may appear that the host computer and HMI are responsible for EMPOP function. However, it is the control layer, via the controller and its low-level routines, that executes all robotic tasks – from directing and monitoring the control of EMPOP processes to actuating single I/O channels – in real-time.

The control routines were written using MP, an embedded programming command language specific to the 6K8 motion controller, and the only programming language supported by it. MP is a low level language; all but the most basic functions, such as toggling outputs or reading inputs, must be coded from the elementary instruction set that comprises the MP language. While MP supports some structured programming features such as "If" statements and "While" loops, it does have some severe limitations[39].

The first limitation is that variables must be assigned to one of 635 32-bit global variables; of these, only 12 real, 10 integer and 10 binary variables can be used to communicate information between the controller and host computer. This is extremely restrictive since the quantity of real-time data that must be exchanged to retrieve sensory information and control the robot exceeds the number of communication variables available. To overcome this, parameters are converted to binary and concatenated together to form single 32-bit binary variables prior to being sent to the robot's control unit. Upon reception, the controller decodes and assigns the data to non-communicating variables. Secondly, since all variables are global, variables must be used with care to prevent accidentally overwriting them. Unfortunately, this limits the usefulness of subroutine calls, since variables must either be dedicated to the subroutine, or their contents must be saved to temporary variables before use. Finally, MP variables cannot be renamed, so descriptive or logical naming conventions cannot be used. Rather, variables must be referred to as VAR, VARI or VARB followed by a number representing each of the real, integer or binary variables, respectively (e.g. VAR1, VARI2 and VARB3).

MP does have one significant advantage over other low-level software development languages. It supports multitasking so that tasks can run concurrently. Since microprocessors have limited memory resources, MP allows a maximum of eleven tasks, including the main or supervisory task, to run in parallel. A programmer can assign any routine to any task. Moreover, any routine in any task can call routines in other tasks. Once a routine is allocated to a task, program flow continues sequentially within that

routine. The only limitation with multitasking is that since all variables are defined globally, a routine cannot be called until any previous instance of it finishes. Otherwise, the cohesiveness and outcome of the routine could be jeopardized as variables were suddenly assigned new values. Semaphores were incorporated into the control routines to prevent this situation from occurring[40].

The most significant programming lesson learned from working with MP was the importance of developing reusable, modular routines. Reusing code reduced the time and microprocessor memory required to develop and implement control algorithms, whereas creating modular routines, with specific functions, made debugging easier. Consequently, the low-level routines were organized in a top-down structure; routines in each layer can only call routines in the same or lower layers. The layer assigned to a routine depended primarily on the routine's role within the EMPOP system; from top to bottom, the routines become more specialized.

To facilitate low-level programming development, control routines specific to each GUI were organized into separate top-down structures. This was possible since the routines associated to a particular GUI were never called by the routines associated to another. The routines that are associated with the Process Monitoring GUI are the most complex and numerous. The coordination and integration of these routines enables the robot to execute the two main sample preparation protocols. Figure 4.8 shows a block diagram of the top-down structure of the control routines associated with the Process Monitoring GUI. Programming the HMI to only call first-order routines ensures separation between the operator-level and the control-level. The first-order control routines are responsible for controlling and governing the system's five main processes: (1) plate cooling, (2) sample filtration, (3) sample ejection, (4) fluid handling, and (5) plate manipulation processes. Conversely, routines at the seventh and eighth levels perform very specific tasks such as filling the pressure reservoir with a volume necessary to generate the specified level of pressure, or energizing valves.

The first-order routines are aided in their tasks by two types of lower level control routines. The first type, shown in grey (Fig. 4.8), are activated by the controller and generally run independently of the main process; the second type are dependent and respond only when called by higher-order programs. Both types of routine will be discussed in the upcoming sections.

4.4.1 Independent control routines

In MP, an advisory program tells the controller when to initiate a particular routine. This advisory program is a special type of MP program that mimics, in software, the function of a Programmable Logic Controller (PLC)[40]. PLC modules are used to simplify the control of automated devices. They are microprocessor-based devices that handle events defined by a change in state of any I/O channel. Typically, the PLC uses logic gates to consolidate all I/O channels into a single line so that it does not have to poll every I/O line individually. When a change of state is detected, the PLC triggers the execution of the user-defined routine associated with that channel. In MP, PLC programs are responsible for I/O scanning and variable monitoring. Consequently, the PLC program generally consists of several "If" statements with different I/O and variable criteria. Moreover, due to MP's multitasking feature, the routines called by the PLC program run as separate tasks in the background.

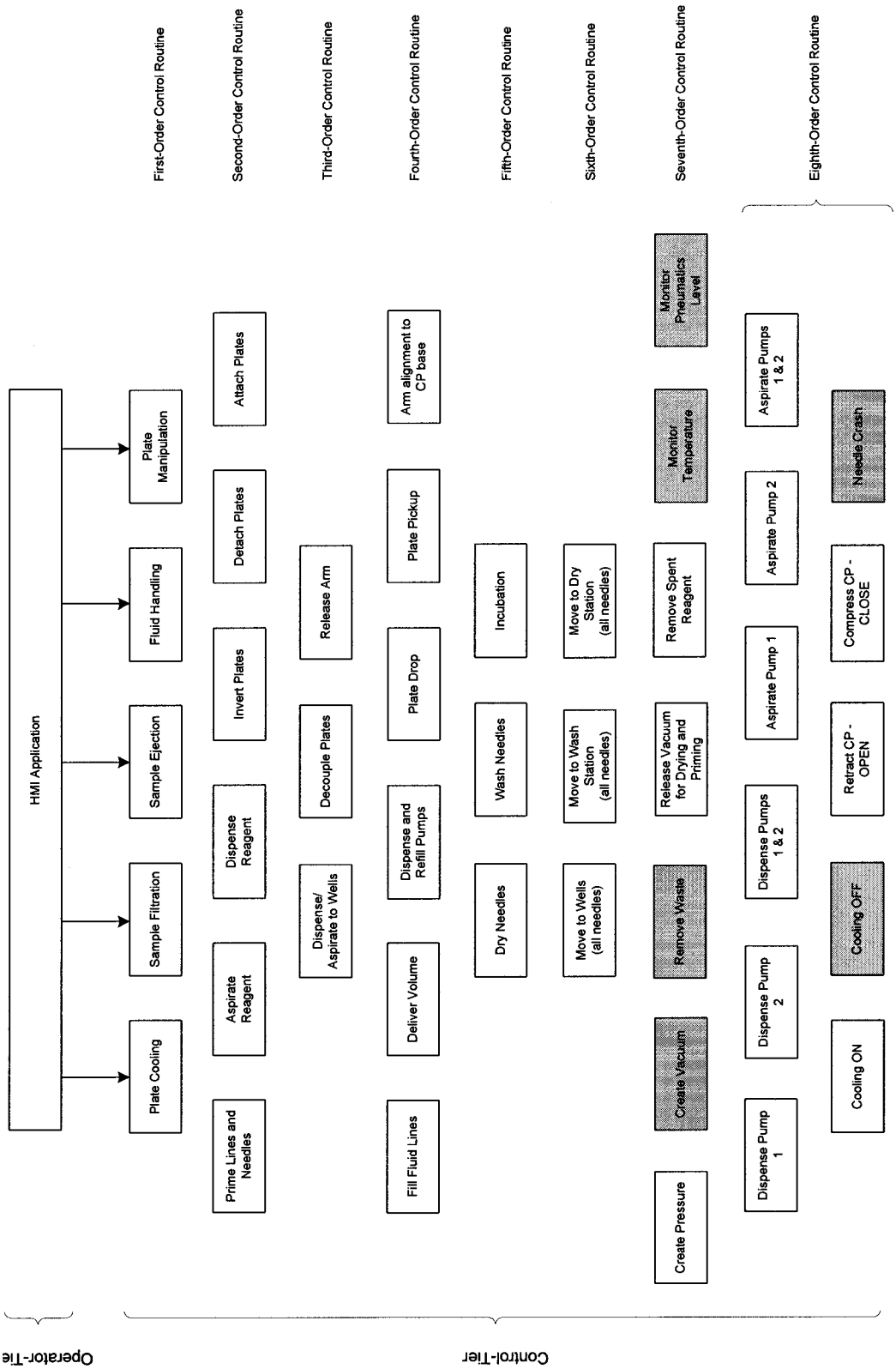


Figure 4.8: Block diagram of top-down organization of control routines associated to the Process Monitoring GUI

The EMPOP system incorporates a PLC program into its software framework for greater efficiency; this allows the controller to perform several functions simultaneously and independently of the routine that is currently executing. Figure 4.9 is a state diagram of the EMPOP robot's PLC program; it activates one of six different control routines in response to specific I/O channels and/or variable conditions. Each routine is dedicated to a different task, and therefore can run in parallel with any other routines. It is important to note that these routines are not strictly bound to the PLC program; they can be called by any same or higher order control routine.

Create Vacuum routine

This routine removes air from the vacuum reservoir to create a pre-defined amount of vacuum. It was included as the first task within the PLC program because it is called repeatedly throughout the protocol. The routine becomes active when the motion controller determines there is a deficiency of vacuum by measuring the vacuum transducer, an analog sensor coupled directly to the vacuum reservoir. As seen in Chapter 3, vacuum is necessary for filtration, aspiration of spent reagent, needle drying and priming. Consequently, this routine cannot execute while the vacuum source is servicing another routine. As a result, the controller uses a `pneumatics semaphore`, a binary variable that is shared amongst all routines requiring vacuum, to indicate the state of vacuum use. Before any of these routines can begin their tasks, they must wait for the `pneumatics semaphore` to become available and then take possession of it for their entire duration.

Remove Waste routine

This routine eliminates system waste that accumulates in the robot's vacuum reservoir. It was tagged to run in the background since it takes a significant amount of time to complete and must occur intermittently throughout processing. Even though waste removal does not depend on vacuum to function, it cannot run in parallel with the *Create Vacuum* routine, as it requires full control of the vacuum reservoir to perform its task. Consequently, the *Remove Waste* routine must first assess the `pneumatics semaphore's` availability prior to executing. In addition, it uses the `waste variable`

to determine whether the system has accumulated enough waste for removal. Although the inclusion of the `waste variable` made programming more complex, it reduced the overall time the robot spent removing waste, allowing it to complete its other tasks with minimal disruption and greater efficiency.

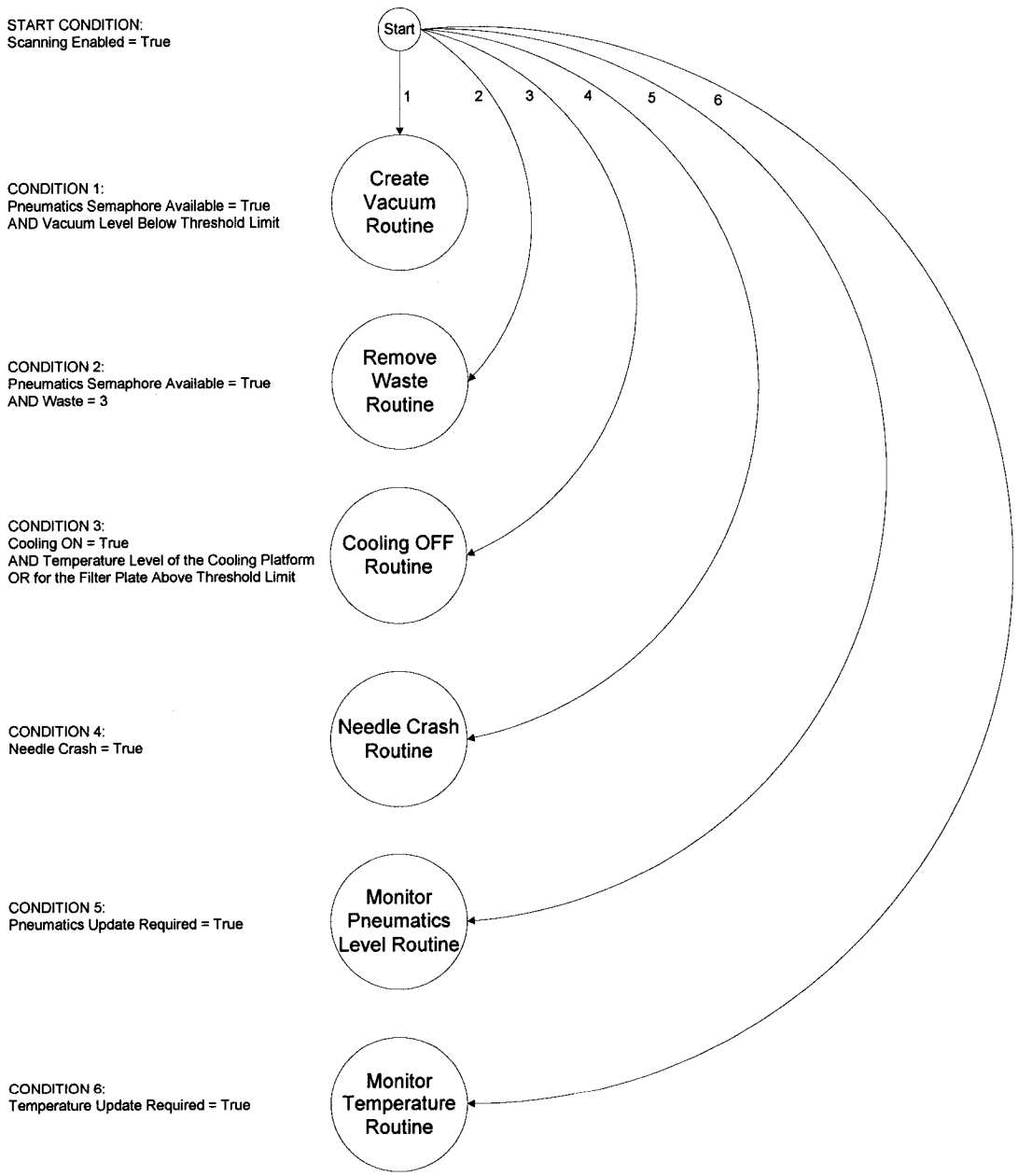


Figure 4.9: State diagram of the PLC program and the routines it activates

Cooling OFF routine

This routine disables the cooling mechanism if the temperature of the CP or filter plate exceeds 35°C. To determine the temperature the controller converts the analog voltage signal emitted from the system's temperature transmitters into degrees. When called by the PLC program, this routine acts as a safety measure to prevent protein denaturation in the unlikely event that a cooling mechanism malfunction results in plate overheating.

Needle Crash routine

This routine causes the robot's 5-axis motion control system to return to its home position; this minimizes damage to the needles during a collision between any of the needle manifolds and the core mechanism or clean station, and simplifies re-calibration. The controller and PLC program detect collisions using magnetic positioning sensors attached to each of the three needle manifolds.

Monitor Pneumatics Level routine

This routine monitors the vacuum and pressure levels within the vacuum and pressure reservoirs, respectively. The data collected from this routine is plotted by the HMI to give the operator a visual indication of the system's pneumatics levels. The PLC program initiates this routine every 500 milliseconds for the duration of a run.

Monitor Temperature Level routine

This routine forwards the temperature of the CP and the filter plate to the Process Monitoring GUI and cooling log file. A new set of temperature readings are sent to the HMI every 2.5 seconds, when the PLC program triggers the execution of the routine.

4.4.2 Dependent control routines

At this point, we return to the robot's five main processes and explain the control algorithms associated to and developed for each. To facilitate the discussion, we will follow the sequence of events that take place from the time the operator initiates a particular step in the Process Monitoring GUI, to the point when the robot declares the task complete.

Plate Cooling

Plate cooling is not included as a separate step within the automated protocol. This is because cooling must occur in parallel with all other processes. Consequently, the operator can activate cooling at any time during the setup procedure. This is advantageous since it takes a substantial length of time for the temperature of the plates and the CP to decrease; if actuated early enough the plates will be cooled to 4°C by the time the samples are loaded into the wells.

Figure 4.10 is the flowchart of the process's first-order control routine. This routine is relatively straightforward and only involves the execution of two eighth-order control routines – *Cooling ON* and *Cooling OFF*. The controller activates this first-order routine when it receives the operator's request to either initiate or terminate plate cooling. These two lower-level routines result in the controller sending either a digital high or digital low signal to a single I/O channel responsible for cooling.

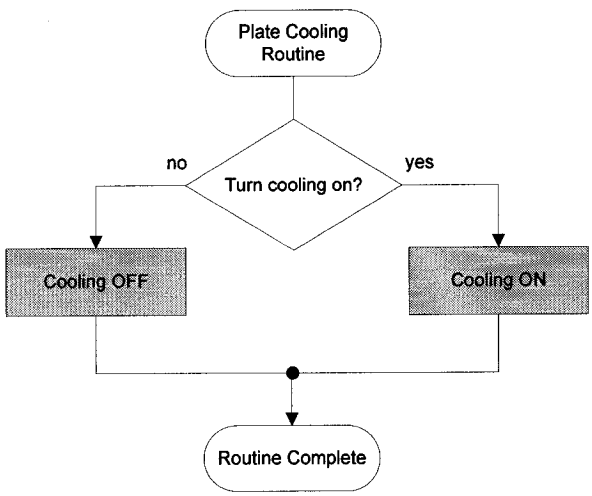


Figure 4.10: First-order control algorithm for the *Plate Cooling* routine. The shaded bases indicate the lower-order routines that are called by the *Plate Cooling* routine

Sample Filtration

Typically, sample filtration occurs only once during a run. However, since the robot is a prototype and its operation still requires optimization, the Process Monitoring GUI allows an operator to conduct many filtration sequences using various levels of vacuum.

Due to the extremely delicate nature of the samples, sample filtration is a very sensitive process. Consequently, the operator must determine the correct balance between filtering the samples too gently or too harshly. Filtering too gently will result in the loss of samples, through reagent aspiration, due to the samples' inability to adhere to the filter membrane, while filtering too harshly will damage the samples' fragile structure.

The sample filtration process is controlled by the parameter settings in the Process Monitoring GUI, shown in the screen capture (Fig. 4.11, Filtration Stage, step 1). This example calls for one repetition using a vacuum of 10.3 mmHg, corresponding to emptying the vacuum reservoir until the analog vacuum transducers reads 4V.

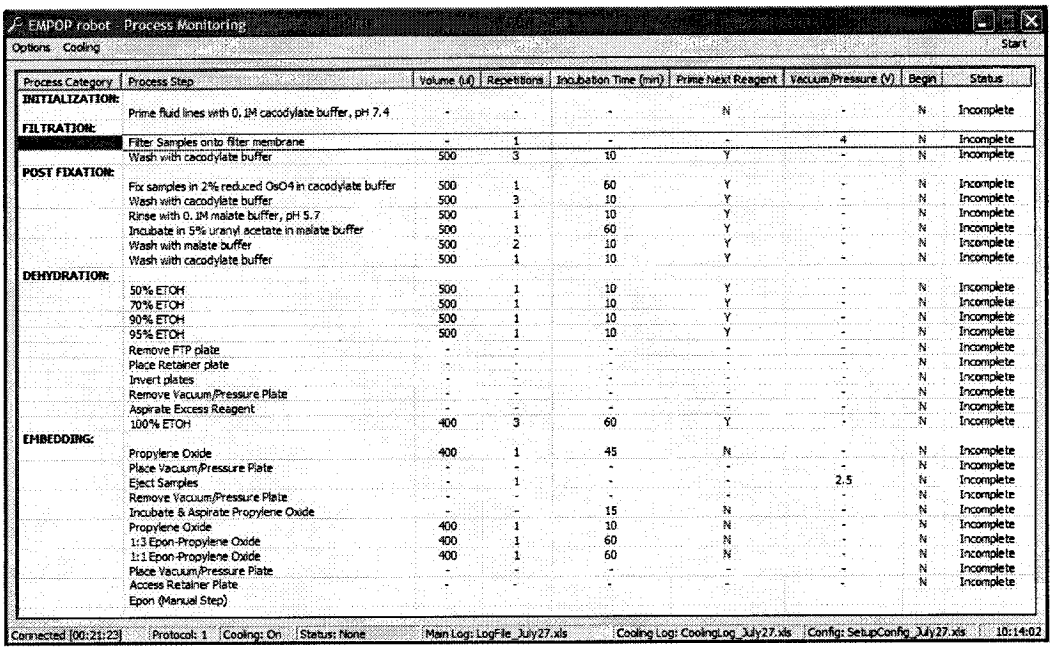


Figure 4.11: Process Monitoring GUI highlighting the filtration step

Sample Filtration routine

Figure 4.12 is a flowchart of the sample filtration's first-order control routine. This routine only calls the *Create Vacuum* routine to perform its task.

Algorithm Description:

(1) PLC program scanning of the *Create Vacuum* and *Remove Waste* routines is disabled. The `pneumatics semaphore's` availability is examined. If it is free, the *Sample Filtration* routine sets the semaphore; if not, it waits for the routine currently using the vacuum source to complete its task and relinquish the semaphore. This is necessary because the level of vacuum in the vacuum reservoir decreases during sample filtration. If pneumatics scanning were left enabled, the PLC program would activate the *Create Vacuum* routine once the vacuum level dropped below the minimum threshold limit. This would cause a significant disturbance to the samples, as they would suddenly begin to experience an increasing force. Moreover, the operator would not be able to filter at a voltage level lower than the threshold limit, because the PLC program would constantly attempt to increase the level of vacuum in the cylinder.

(2) Any residual vacuum left in the reservoir is drained. This may seem to be a waste of resources but it greatly simplified programming, since it avoids comparing the amount of vacuum within the reservoir against the amount that is required.

(3) The *Create Vacuum* routine is called to create the required amount of vacuum in the reservoir. To determine when to cease emptying, the motion controller continually reads the analog channel associated with the vacuum reservoir's transducer until its voltage is 4V or greater. To compensate for the occasional high voltage spike on the analog line, the controller uses the average value of ten transducer readings to monitor the level of vacuum in the reservoir.

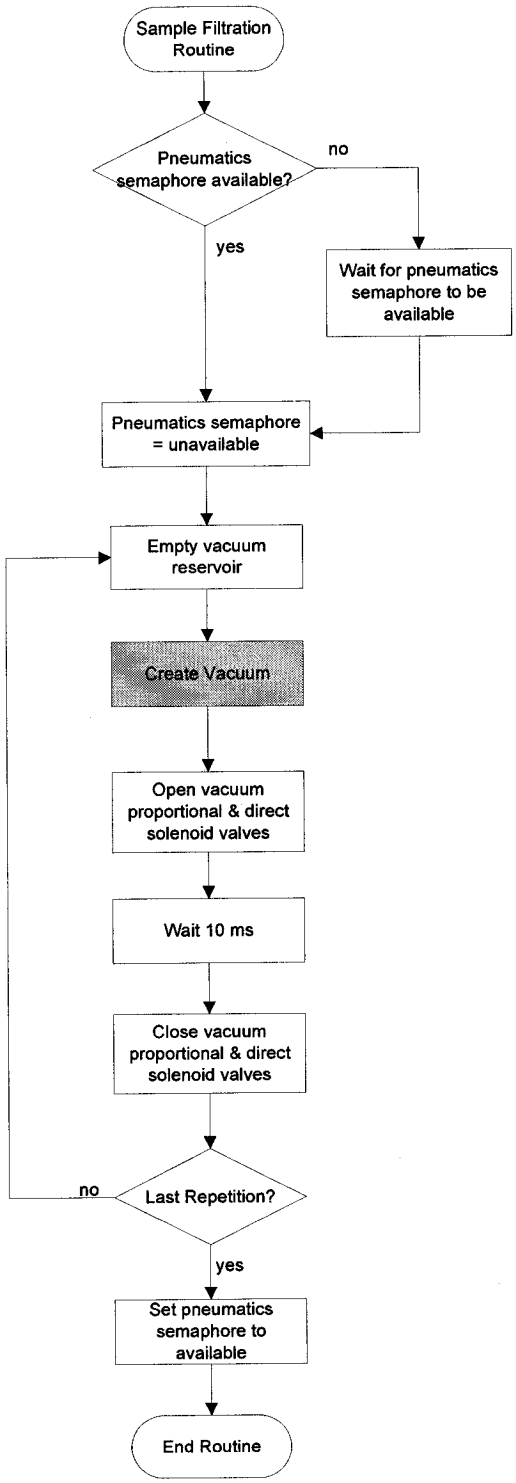


Figure 4.12: First-order control algorithm for the *Sample Filtration* routine. The shaded base indicates the lower-order routine that the *Sample Filtration* routine calls.

(4) The vacuum source is then released into the VPP, by actuating the proportional and direct solenoid valves.

(5) These valves remain open for a period of 10 seconds.

(6) The motion controller closes the valves, once the timer expires.

(7) If further repetitions are required the controller loops back to the beginning of the algorithm. Otherwise, it continues at step 8.

(8) The `pneumatics semaphore` is declared available.

(9) The sample filtration process is deemed complete.

Sample Ejection

Sample ejection is very similar to sample filtration and like filtration it generally occurs only once during the protocol (Fig. 4.13). The main difference between ejection and filtration is in the Operator Tier. In addition to setting the number of repetition sequences and pressure level, the operator can also specify the number of pressure bursts and the duration of each burst for a single pressure release. These two additional parameters are set via a dialog box in the Process Monitoring form's Options menu (Fig. 4.14). These parameters were included because ejection experiments indicated that several low-pressure pulses were more effective then a single high-pressure pulse in detaching the samples from the filter plate's membrane/screen.

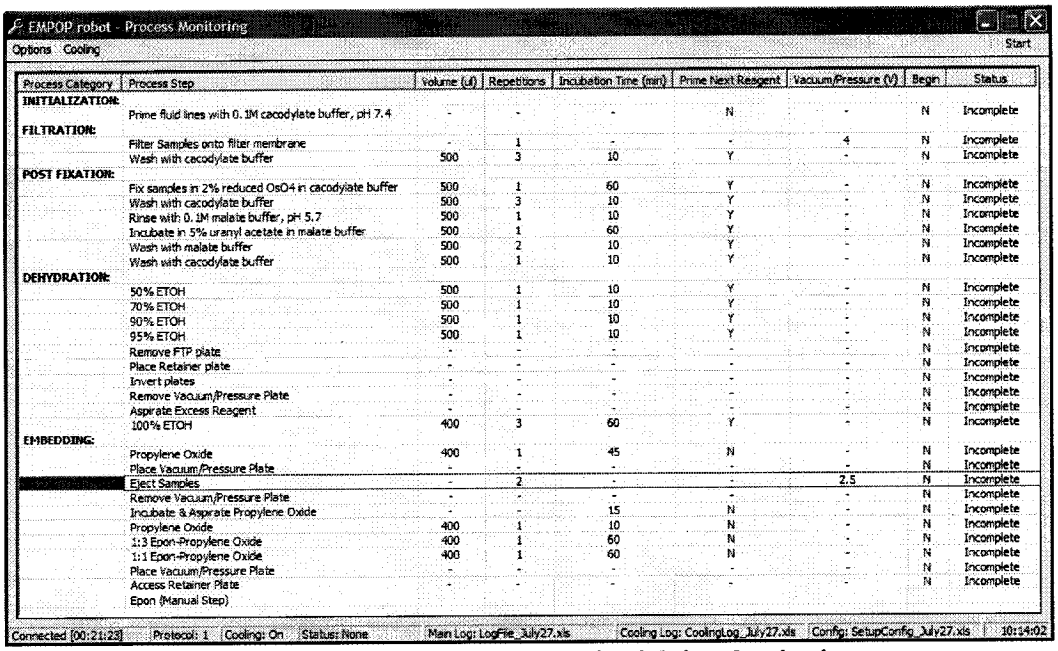


Figure 4.13: Process Monitoring GUI highlighting the ejection step

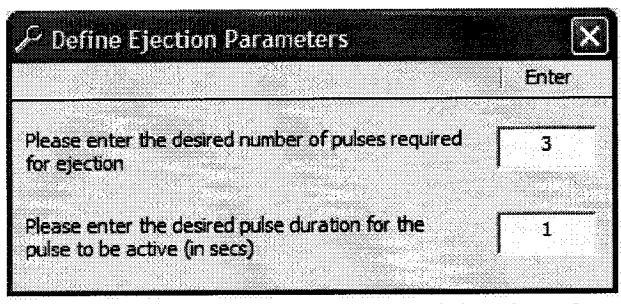


Figure 4.14: Ejection parameter menu for defining the pulse sequence

Sample Ejection routine

This section describes how the ejection algorithm, shown in Figure 4.16, differs from the filtration algorithm. This routine will be described for the sample parameters shown in the Process Monitoring GUI (Fig. 4.13, Embedding Stage, step 3), which calls for two sequences starting with a pressurization level of 50 psi, corresponding to a pressure transducer voltage level of 2.5V. As shown in Figure 4.15, each pressure release sequence will consist of three pulse cycles with an active pulse duration of 1 second.

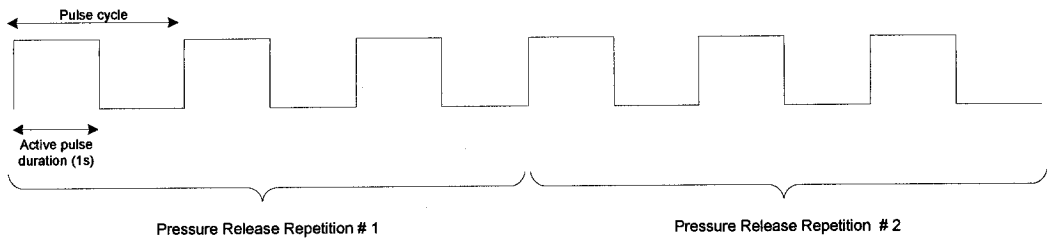


Figure 4.15: Pulse sequence corresponding to operator defined parameters for ejection step

Algorithm Description:

(1) Pneumatics scanning is disabled prior to filling the pressure reservoir. This is because the *Create Vacuum* and *Create Pressure* routines use the same compressor to perform their task. Consequently, the system cannot create both vacuum and pressure simultaneously.

(2) The pressure reservoir is emptied.

(3) The *Create Pressure* routine is called to fill the pressure reservoir to a level corresponding to a transducer reading of 2.5V. As with the *Create Vacuum* routine, pressurization is deemed complete when the average of the last ten pressure transducer readings is equal to or greater than 2.5.

(4) The motion controller repeatedly actuates and de-actuates a series of valves for a period equal to the active pulse duration. Although each pulse train is identical, the actual pressure delivered to the plates, via proportional and direct pressure valves, decreases since the pressure in the reservoir drops following each pulse cycle. This has the desired effect of allowing the samples to gently detach from the filter membrane/screen, and thus prevents damage to their fragile structure.

(5) After the first pulse train is completed, the robot loops back to step 4 to execute the second pulse train.

(6) The sample ejection process is complete.

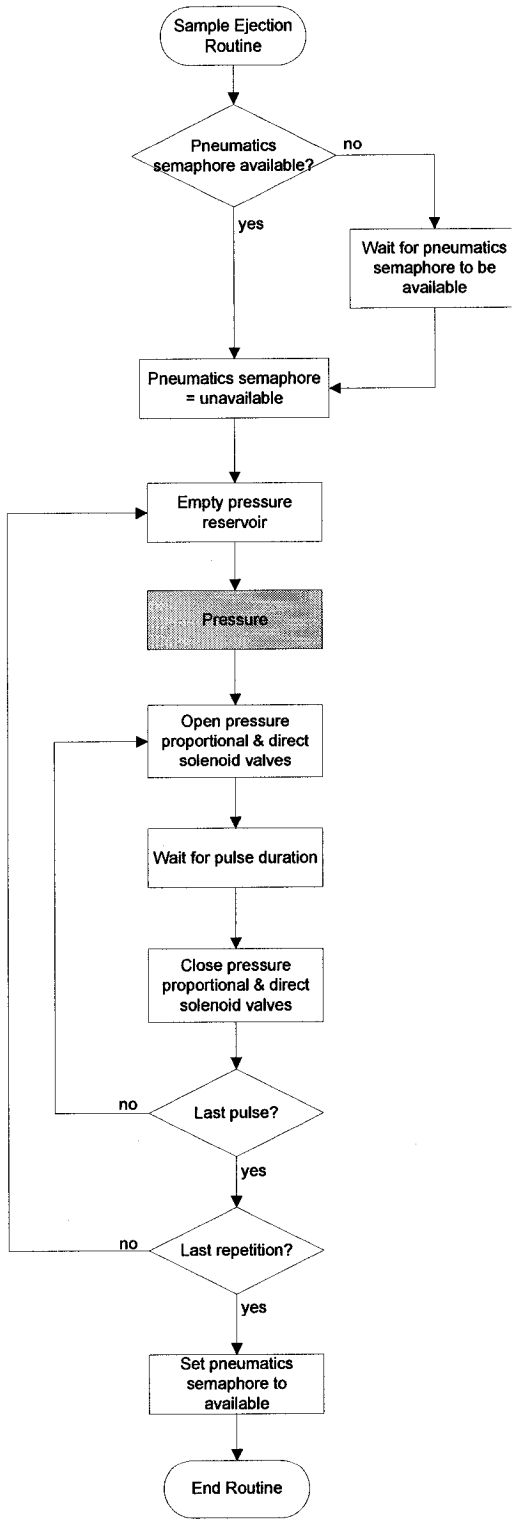


Figure 4.16: First-order control program for the *Sample Ejection* routine. The shaded base indicates the lower-order routine that is called by the *Sample Ejection* routine.

Fluid Handling Process

The fluid handling process is responsible for the execution of approximately 63% of the steps within the protocol. Figure 4.17 highlights these steps within a screen capture taken of the Process Monitoring GUI. Figure 4.18 shows the routines involved in this process. It is the most complex process to automate since it requires the integration of many routines within different layers to process each well.

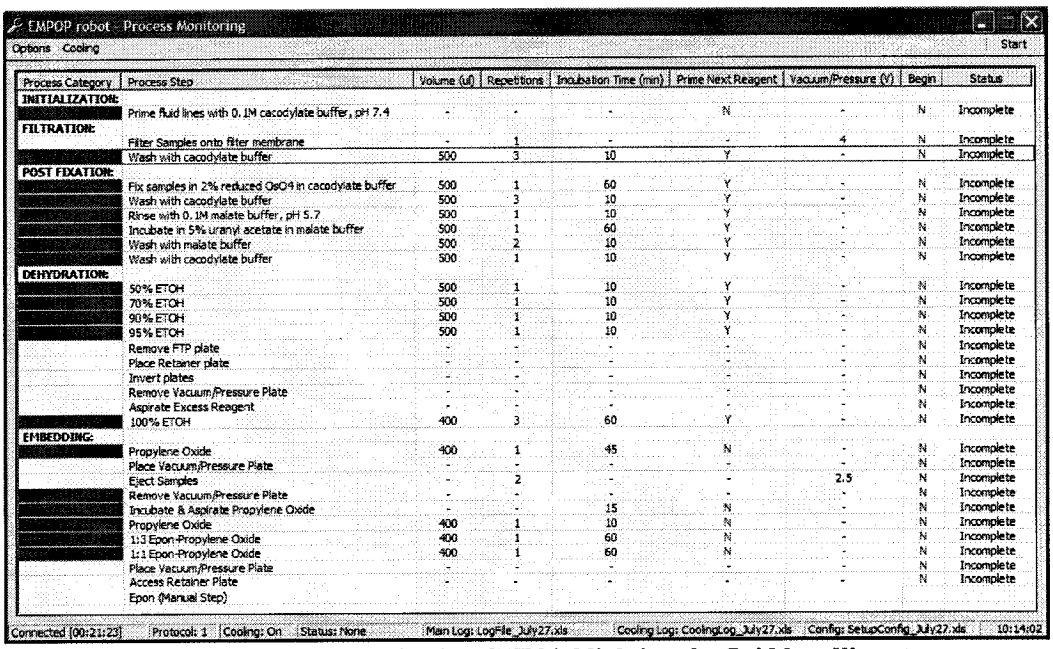


Figure 4.17: Process Monitoring GUI highlighting the fluid handling steps

To explain this process, we will use the washing samples with Cacodylate buffer step (Fig. 4.17, Filtration Stage, step 2) as an example. The parameters specific for this step require the robot to wash each sample 3 times with 500 µl of buffer. In addition, the samples are incubated for 10 minutes between each wash cycle. To increase efficiency, the operator has requested that the system prime the fluid lines with the next reagent (Osmium Tetraoxide) during the last incubation period.

Once the HMI receives confirmation that the control unit has downloaded these parameters into memory, it calls the system's first-order control routine for fluid handling. The algorithm used to implement this routine is outlined in Figure. 4.19.

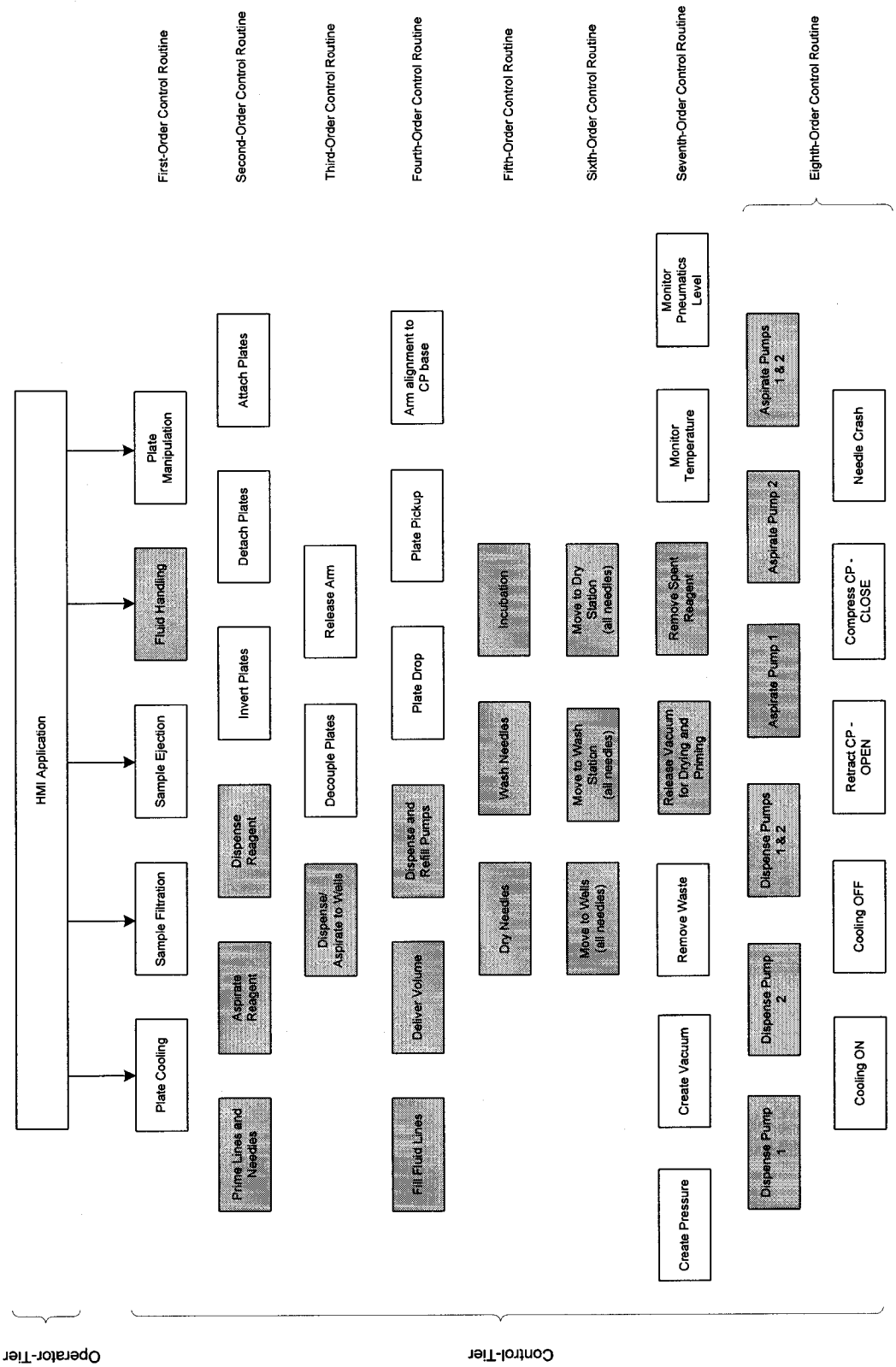


Figure 4.18: Top-down organization of control routines indicating the routines associated with the fluid handling process.

Fluid Handling routine

To accomplish its task, this routine calls on four lower-order routines: (1) *Priming Lines and Needles, (2) Dispense Reagent, (3) Aspirate Reagent, and (4) Incubation.* Routines 1, 2 and 3 will be discussed in separate sections, as they themselves call on several lower-order routines to complete.

Algorithm Description:

(1) The process begins by determining if the fluid line has been primed with the reagent to be dispensed. This information comes from the HMI application, which tracks which reagent the fluid lines are currently primed with. For our example, the line has been primed since the first step within the protocol (Fig. 4.17, Initialization, step 1) is to prime the lines with Cacodylate buffer.

(2) The robot must then ascertain whether the step requires it to dispense a reagent to the wells. This is necessary because certain steps in the protocol only call for priming and/or aspiration. For example, when the embedding plate configuration is first formed (Fig. 4.17, Dehydration, step 9), the robot must remove, via the aspiration needle, any excess reagent that may have accumulated on the reverse side of the filter plate due to filtration and/or previous reagent processing steps.

(3) In this example, the robot must dispense 500 µl of Cacodylate buffer to each well being processed. The *Dispense Reagent* routine, a second-order routine, controls reagent dispensing.

(4) The robot incubates the samples in Cacodylate buffer for 10 minutes with the *Incubation* routine. Sample incubation is defined within a separate routine so it can run in the background, allowing the robot to complete other tasks, such as priming the fluid lines with the next reagent or expelling waste from the vacuum reservoir, while the samples are incubating.

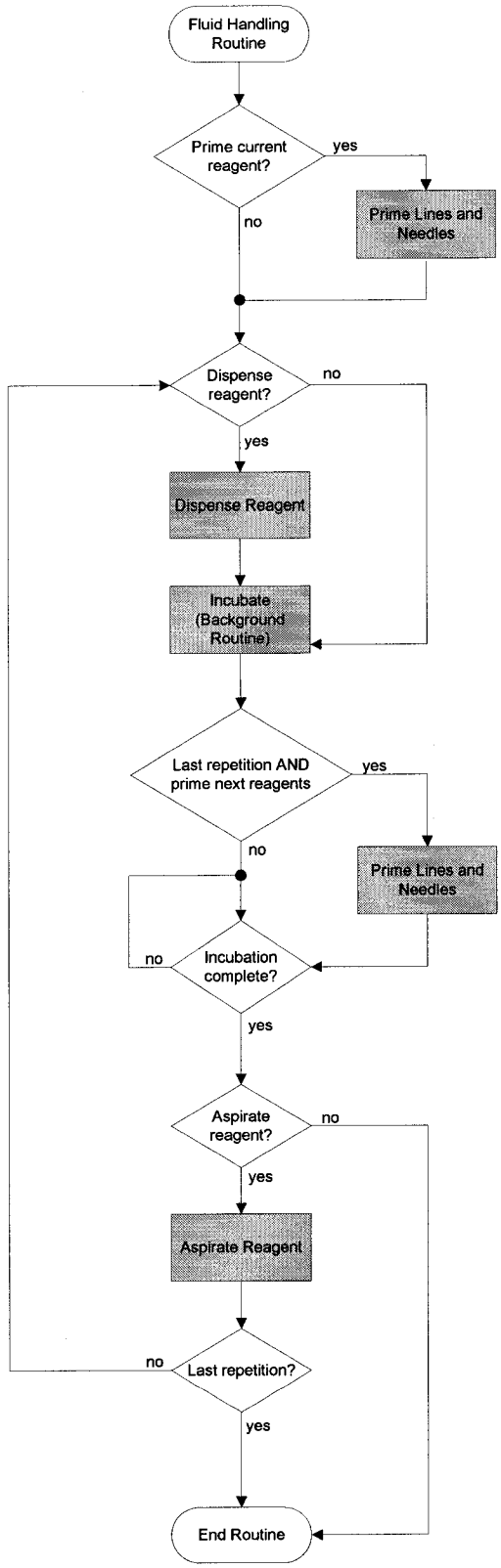


Figure 4.19: First-order control algorithm for the *Fluid Handling* routine. The shaded bases indicate the lower-order routines that are called by the *Fluid Handling* routine.

(5) If the robot is on its last processing repetition, it primes the fluid lines with the next reagent during incubation. Within the context of our example, priming may only occur after the first two dispense and aspirate sequences are complete. The *Priming Lines and Needles* routine accomplishes priming. The advantage of priming the lines during incubation is that it prevents the samples from drying out, since otherwise the system would have to prime the fluid lines at the start of the next automated step. This could take considerable time if mixing is required.

(6) Using the *Aspirate Reagent* routine, the robot proceeds to aspirate reagent for each channel.

(7) When processing is complete the controller will send a process completion notification to the HMI, which will update the step's status column to complete. Otherwise, the robot will loop back to the beginning of the control algorithm until all repetitions are executed.

Priming Lines and Needles routine

Priming is the act of filling the system's fluid lines, from reservoir to needle, with the reagent being dispensed. By doing this, the robot removes unwanted reagent left in the lines from the previous step. Although the general concept behind priming is simple, automating this function was complicated because it requires the precise coordination and actuation of many different valves. Priming is particularly difficult to achieve when reagents must be mixed.

To prime the lines, this routine calls ten lower-order routines. Due to their complexity, the *Fill Fluid Lines* and *Dispense and Refill Pumps* routines will be dealt with separately; the remaining eight routines will be explained as they appear within the discussion.

Algorithm Description:

(1) The needle is moved to the robot's dry station where priming takes place. The dry station is used because it can easily displace unwanted reagent collected in it to the vacuum reservoir with the actuation of a single valve. Positioning is achieved by the *Move to Dry Station* routine, which uses the pointer-data structure, defined during system calibration, to determine where to move the robot's 5-axis system.

(2) The robot's two variable volume stepper pumps begin pumping fluid through the Y-axis fluid handling panel's network of valves and fluid lines into the robot's dry station via the *Fill Fluid Lines* routine. Reagent priming is complete when the *Fill Fluid Lines* routines finishes.

Fill Fluid Lines routine

To control reagent priming this routine must simultaneously perform two separate yet highly important tasks.

Task 1: This routine calculates and monitors the number of aspirating and dispensing full-strokes that are required to prime the line and needles. This was necessary since the pumps cannot push more that 2560 µl per stroke, whereas the volume needed to prime the fluid lines is ~30 ml. Experiments showed that 12 aspirating and dispensing full-stroke sequences (i.e. 12 aspirating full-strokes and 12 dispensing full-strokes) were required to prime the lines with most processing reagents. Propylene Oxide requires 18 full-stroke sequences since it has a tendency to fragment in the lines. Conversely, Epon-Propylene Oxide solutions require 24 full-stroke sequences because of their high viscosity. Once the fluid lines are primed, the system must ensure that the needles are primed as well. This situation only arises if the robot will be delivering reagent from more than one needle at a time. This is because the start needle automatically gets primed while the fluid lines are priming. Priming the additional needles requires the robot to perform 2 additional aspirating and dispensing full-stroke sequences for each needle.

The situation is more complex when mixing is required. To understand why, consider how the pumps achieve mixing; both pumps push fluid through a mixer whose serpentine or spiral geometry passively mixes the reagents. To assure that reagents are completely mixed to form a homogenous solution, both reagents must arrive at the mixer at the same time. The sample preparation protocol rarely requires mixing of reagents in equal proportions, so when mixing the routine must command one pump to push more liquid than the other. However, if both reagents are to arrive at the mixer at the same time, this routine must also increase the pumping velocity of one pump compared to the other.

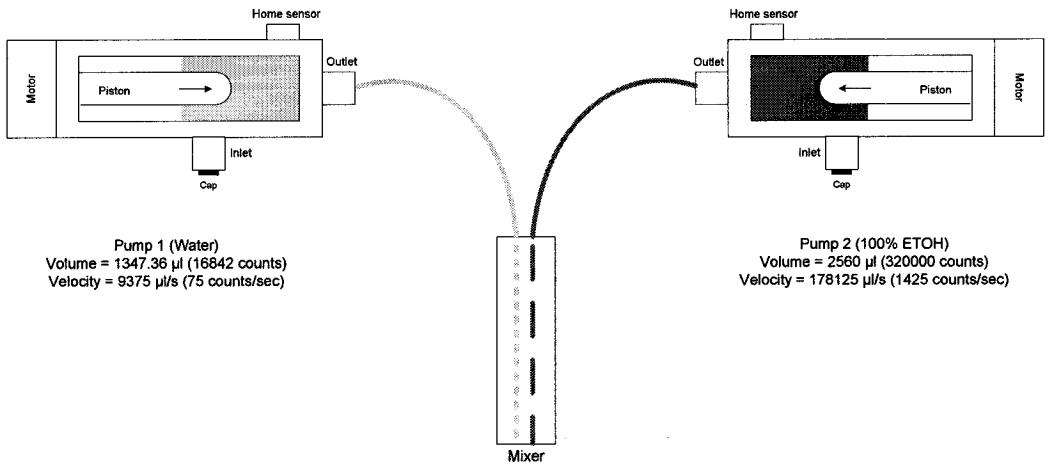


Figure 4.20: Schematic illustration of mixing for the production of 95% ETOH

Figure 4.20 is a schematic illustration of how the pumps work to produce 95% Ethanol by diluting 100% Ethanol with water. The ratio of water to 100% Ethanol required to produce the desired dilution is 1:19. Since the flow rate for each pump is directly proportional to its ratio, it is clear that in this case, Pump 1 must pump a smaller volume of water at a slower rate than Pump 2, to guarantee that the reagents reach the mixer at the same time. This routine calculates the flow rates for each pump using the ratio. There are two main disadvantages with this method of priming the lines for mixing.

The first disadvantage is that the line is only considered primed when the reagent being dispensed in the smaller proportion reaches the needles. For example, when mixing 95% Ethanol, priming is only complete when water reaches the needles. Consequently, instead of requiring 12 full-stroke aspirate-dispense sequences to prime, the system requires 228 full-stroke sequences. This value was calculated by multiplying the number

of full-strokes required, 12, by the ratio, in this case 19. This results in significant wastage of 100% Ethanol. To reduce wastage, the flow rates are only adjusted once the reagents reach the mixer, which was found to take 3 full-stroke sequences. This results in a full-stroke reduction of ~24%, as the robot only requires a total of 174 full-strokes to prime. This value was calculated by determining the number of full-strokes required to reach the needles from the mixer, (12-3)*19 = 171, and then adding the number of full-strokes required for the reagent to reach the mixer from the fluid reservoir, 171 + 3 = 174. The second problem is a direct consequence of the first and is the time required to prime the lines. As noted above, the mixing algorithm requires that the pump delivering the reagent in smaller proportion must use a lower velocity. As a result, it takes longer for the priming sequence to complete. These disadvantages intensify as the mixing ratio between the two reagents increases.

Several different approaches to mixing were tested in an attempt to minimize these problems. Unfortunately, the best remedy found required modifying the fluid handling panel's hardware to incorporate an active mixer with a temporary fluid storage chamber. However, as the robot is still a prototype and since these problems only pose trouble when mixing reagents in very small or large proportions, it was decided to address this problem later. In the interim, the problem can be avoided, by manually preparing a mixed reagent and choosing a reagent line for the solution.

Task 2: Since the robot's dry station can only hold a small volume of liquid, this routine must ensure that the dry station does not overflow during priming. It achieves this by calling the *Release Vacuum for Drying and Priming* routine after every third full-stroke sequence. Once initiated, this eighth-order control routine actuates a valve to draw unwanted reagent from the dry station to the vacuum reservoir. Additionally, it updates the system's `waste variable`. Doing so will cause the PLC program to activate the *Remove Waste* routine. Unwanted reagent cannot be moved into the vacuum reservoir while waste removal is in progress since both routines require sole control over the vacuum reservoir to function. Consequently, priming must be temporarily halted while the *Remove Waste* routine is running.

Dispense and Refill routine

Figure 4.21 is a schematic illustration of the pumps during filling and dispensing. To fill the pumps, the robot must simultaneously open the inlet valve and close the outlet valve. This is achieved by the *Aspirate Pump 1* and *Aspirate Pump 2* routines. Mixing requires the execution of a separate routine, *Aspirate Pumps 1 & 2*, which opens and closes the inlet and outlet valves for both pumps. In a similar manner, emptying the pumps entails closing the inlet valve while simultaneously opening the outlet valve (*Dispense Pump 1*, *Dispense Pump 2* and *Dispense Pumps 1 & 2)*. Once the pumps finish dispensing, this routine refills each pump. At first glance, from the standpoint of mixing, it is not obvious why it is necessary to refill both pumps since one pump still contains reagent. However, from a programming standpoint, having both pumps refill and dispense at the same instant for the same amount of time simplified the algorithm by eliminating the need to keep track of an extra looping parameter, and subsequently determine when each pump needed refilling.

While the pumps are refilling, the PLC program calls the *Create Vacuum* routine to replenish the vacuum lost by moving reagent from the dry station to the vacuum reservoir. The advantage of this setup is that unlike waste removal, the creation of vacuum does not require the priming process to halt, thereby improving efficiency.

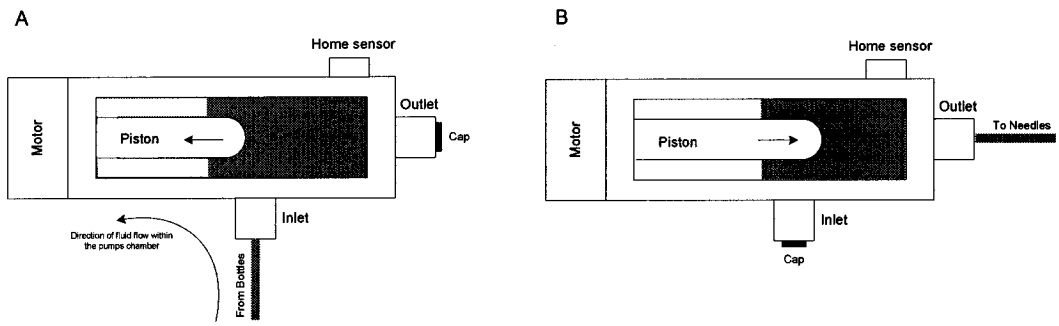


Figure 4.21: Schematic illustration of pump function: (A) pump filling, (B) pump dispensing

Dispense Reagent routine

This routine incorporates and integrates three separate tasks: (1) moving the needles to each well and the clean station; (2) delivering the required volume of reagent to each well; and (3) washing and drying the needles according to the specified washing

frequency. Figure 4.22 shows the flow diagram for this routine. To perform its tasks this routine calls on three lower-order routines: (1) *Dispense/Aspirate to Wells*, (2) *Wash Needles*, and (3) *Dry Needles*. These routines will be discussed in detail once the *Dispense Reagent* routine is explained.

Algorithm Description:

(1) The robot's 5-axis motion control system moves the needles to the wells, where they will subsequently dispense the specified volume of reagent. Needle positioning and reagent dispensing are achieved by the *Dispense/Aspirate to Well* routine.

(2) If the washing frequency selected requires the system to clean the needles between each dispense sequence, the robot will proceed to wash the needles and dry the needles via the *Wash Needles* routine and the *Dry Needles* routine, respectively. Otherwise, the needles will be washed and dried after the robot has delivered reagent to the entire array of wells.

(3) The robot will loop back to execute steps 1 and 2 until all the wells have been processed.

(4) The Z and Y-axes are sent to their respective zero-reference locations prior to sample incubation and/or reagent priming.

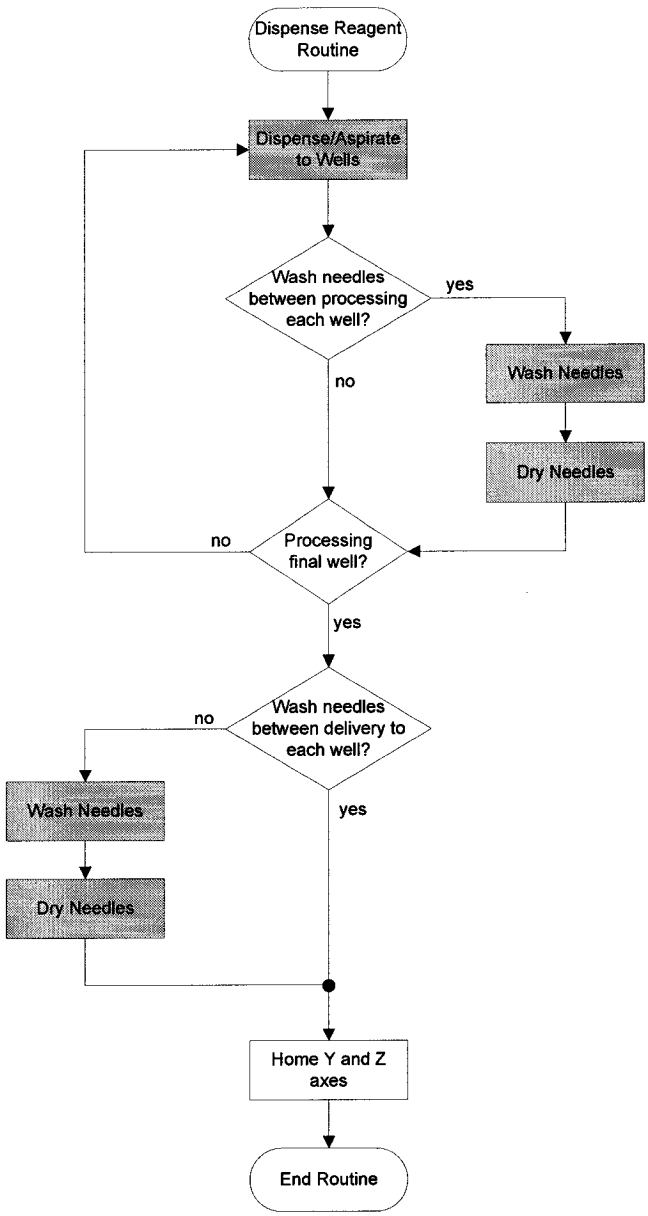


Figure 4.22: Control algorithm for the *Dispense Reagent* routine. The shaded boxes represent the lower-order routines that the *Dispense Reagent* routine calls.

Dispense/Aspirate to Well routine

This routine performs two tasks during reagent dispensing. It can also perform a third task during the reagent aspiration, as will be described later.

Task 1: The needles are positioned to each well using the Move to Wells routine. This routine retrieves the interpolated well locations, defined during calibration, to determine the X, Y, Z and θ coordinates of the wells.

Task 2: The *Deliver Volume* routine dispenses reagent to the wells. This routine is very similar to the *Fill Fluid Lines* routine, described in the previous section. The same lower-order routines are used and the pumps are controlled in the same way. The only difference between these routines is the volume that each pump pumps. During priming the pumps dispense a full-stroke (2560 µl), unless mixing, in which case they dispense a fraction of a full stroke proportional to the ratio of the reagents being mixed. Contrarily, reagent delivery requires the pumps to pump a variable volume specified by the operator. Consequently, before this routine calls the *Dispense and Refill* routine, it must first calculate the fraction of a full-stroke that the pumps must move to deliver the required amount of reagent. To deliver volumes that are greater than the pumps maximum capacity, the algorithm tracks the number of full strokes that are dispensed prior to dispensing the fractional amount. For example, to dispense 2700 µl, the robot first pumps 2560 µl, refills, and then pumps the remaining 140 µl. In addition, the routine also includes another looping variable to ensure that pumps deliver fluid to the wells via all the needles being used.

Wash Needles routine

The *Move to Wash Station* routine actuates the system's two gear pumps, which causes fresh water to fill the station. Once filled to the appropriate level, the robot moves the needles to the wash station, where they are soaked for a duration specified by the operator.

Dry Needles routine

The *Move to Dry Station* routine places the needles inside the dry station, where needle drying takes place via the *Release Vacuum for Drying and Priming* routine. By actuating a valve, this routine creates a direct path for excess water remaining on the needles after needle washing to be pulled into the vacuum reservoir.

Aspirate Reagent routine

The *Aspirate Reagent* algorithm is similar to the *Dispense Reagent* routine. The main difference is that the system removes reagent rather than delivers it. Thus, this section

will focus on the how the *Dispense/Aspirate to Wells* routine achieves aspiration. Earlier, it was mentioned that this routine performs three tasks; two of them (Tasks 1 and 2) were described in relation to reagent dispensing. Reagent aspiration also requires the execution of two of the routine's three tasks - Task 1 and Task 3.

Task 1: As explained in the previous section, the *Move to Wells* routine moves the robot to the required well locations.

Task 3: The *Remove Spent Reagent* routine controls reagent aspiration. This seventh-order control routine removes reagent from each well by actuating a valve, which establishes a direct path between the vacuum reservoir and the aspiration needles so that unwanted reagent is sucked into the vacuum reservoir. To prevent aspirating the samples the valve is only opened for three seconds.

The development of this routine was challenging for two reasons. The first is a direct result of the system having to periodically re-establish vacuum by having the PLC program call the *Create Vacuum* routine when the level of vacuum in the cylinder is below the threshold required for aspiration. Consequently, the *Remove Spent Reagent* routine must first check the status of the `pneumatics semaphore` prior to aspirating. If the semaphore is available, the routine takes possession of it, executes, and then releases it. Otherwise, the routine waits for it to become available prior to executing. The second requires the robot to increment the `waste variable` after every twentieth well is aspirated. This value was chosen because the robot generally removes small amounts and consequently, waste removal does not have to occur as frequently as priming; this minimizes the time required to complete the aspiration sequence.

.

Plate Manipulation

As seen in Figure 4.23, this process takes place during the last part of the dehydration stage, and during the embedding stage. Plate manipulation reconfigures the core mechanism into the delivery and embedding assemblies by interchanging the core mechanism's FTP, VPP or retainer plate, or inverting the assemblies. The Process

Monitoring GUI treats each step in the formation of a new plate configuration as a separate processing step to provide better control over the protocol. If the robot fails to detach, attach, or invert the necessary plate(s), the operator can correct the situation by manually completing the unsuccessful step and, after recalibration, can resume processing. To perform its task this process calls the control routines highlighted in Figure 4.24.

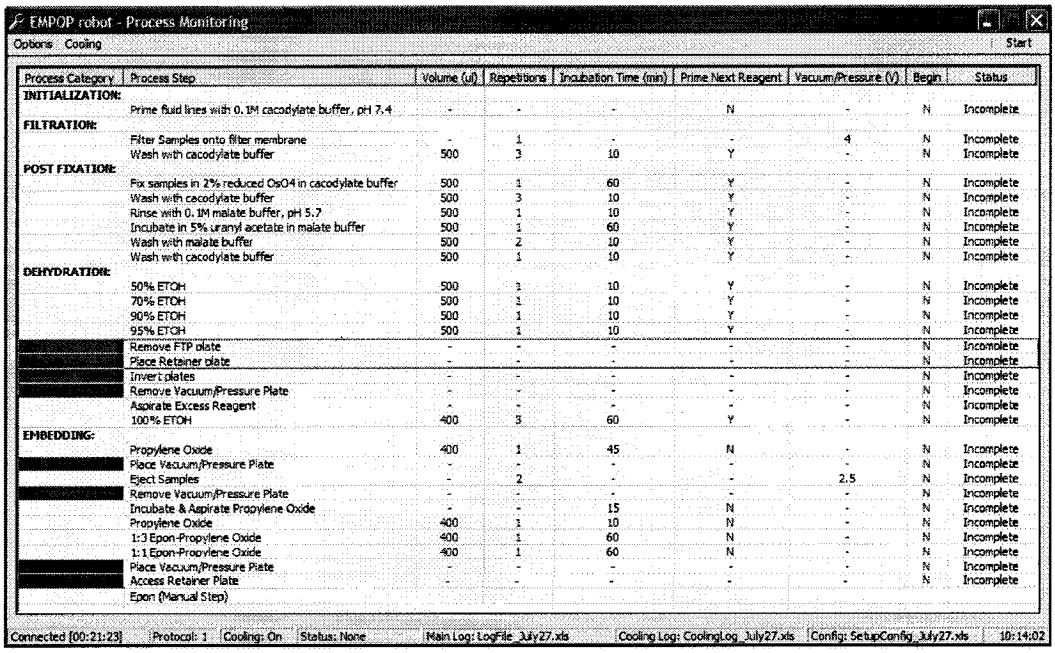


Figure 4.23: Process Monitoring GUI highlighting the plate manipulation steps

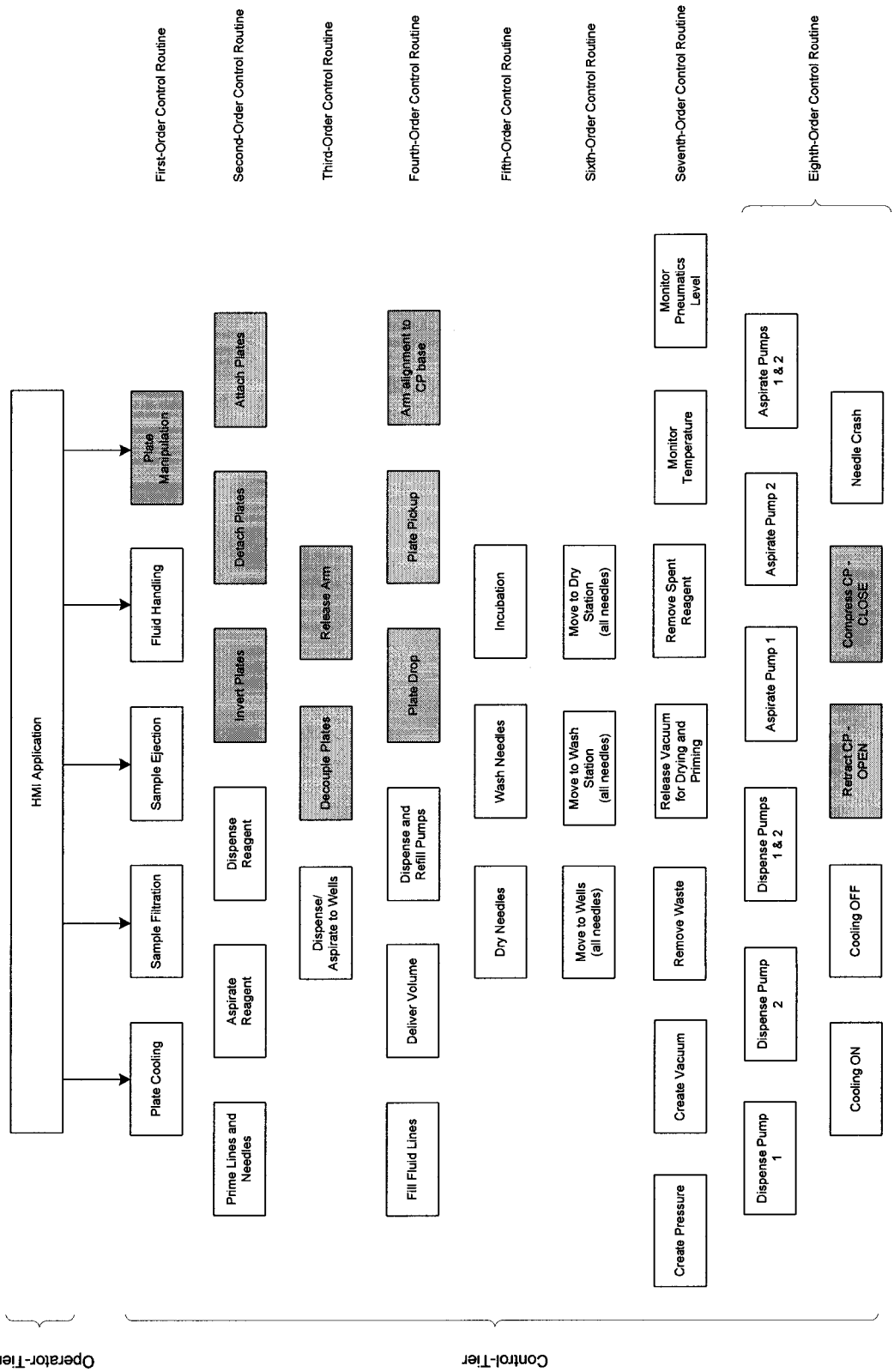


Figure 4.24:Top-down organization of control routines indicating the routines associated with the plate manipulation process.

Plate Manipulation routine

The creation of the different plate configurations involves three steps: (1) removal of the unwanted plate, (2) placement of the required plate, and in some cases, (3) inversion of the plate assembly. Consequently, as shown in Figure 4.25, this routine calls three second-order control routines - *Detach Plates*, *Attach Plates* and *Invert Plates* - which are specific to and responsible for each task. The only information that the Process Monitoring GUI forwards to the *Plate Manipulation* routine is the plate selected for processing. Structuring the first-order control algorithm in this manner allowed for a simpler and modular implementation of the plate manipulation process.

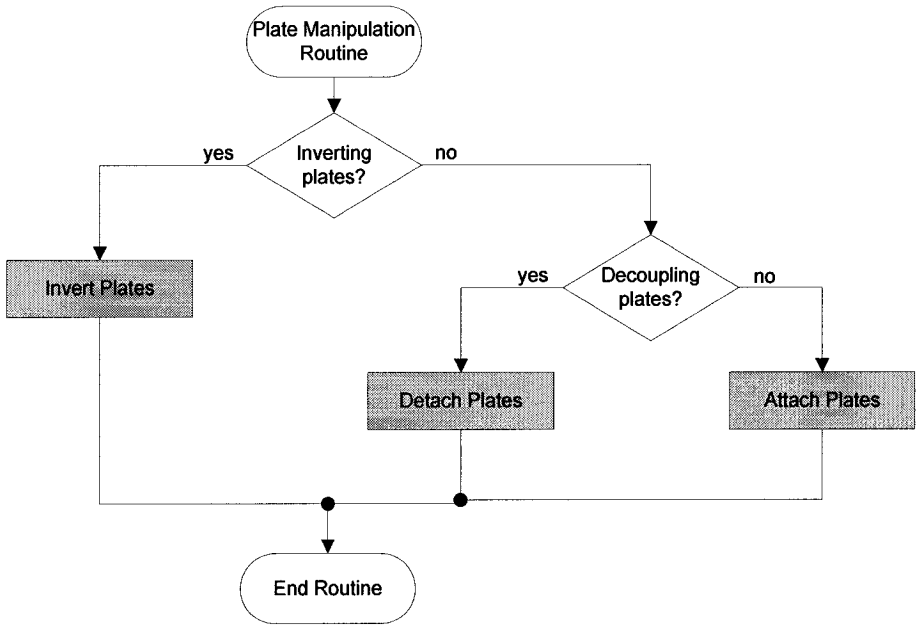


Figure 4.25: First-order control program for the *Plate Manipulation* routine

At the time of writing this thesis, the hardware for the transfer platform was not yet fully installed. Consequently, the code for the *Invert Plates* routine was not complete.

To simplify the discussion, each of these second-order routines will be explained with reference to how the robot establishes the delivery configuration (Fig. 4.23, Dehydration Stage, steps 5 and 6) via the removal of the FTP (*Decouple Plates*) and the subsequent placement of the retainer plate (*Attach Plates*).

Detach Plates routine

Figure 4.26 is the flowchart showing the procedure used to remove and relocate the FTP. Four complex routines are called: (1) *Arm Alignment to CP Base*, (2) *Decouple*, (3) *Plate Drop*, and (4) *Release Arm*. These will be discussed in separate sections.

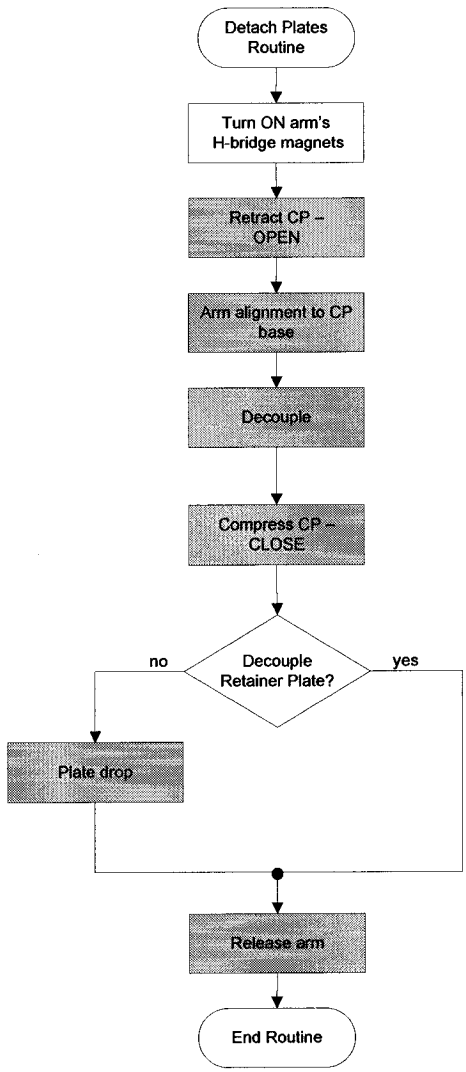


Figure 4.26: Control algorithm for the *Detach Plates* routine. The shaded bases represent the lower-order routines that are called by the *Detach Plates* routine.

Algorithm Description:

(1) Affixing the arm to the plates requires the arm's two locating (dowel) pins to align perfectly to the plates' locating holes. Although system calibration ensures that the location of the plates with reference to the arm is known, actuating the

arm's electromagnets prior to positioning (step 3) aids the system in guiding the arm into the locating holes; the attractive force between the arm's electromagnets and the corresponding steel discs on the FTP cause the arm to attach to the plate before motion ceases.

(2) To give the arm more clearance for attachment and decoupling, the *Retract CP – OPEN* routine retracts the inner walls of the CP. This eighth-order control routine simultaneously actuates the spring mechanism attached to each CP surface.

(3) The robot's 5-axis motion system is moved to the processing configuration where the arm is attached to the FTP. The *Arm Alignment to CP Base* routine aligns the arm to the plates by monitoring the digital proximity sensor located on the electromagnetic arm.

(4) The *Decouple* routine controls plate decoupling and monitors the arm's proximity sensor to ensure that the decoupling sequence has executed.

(5) The *Compress CP – CLOSE* routine closes the CP's inner walls by simultaneously de-actuating the platform's two spring mechanisms.

(6) The *Plate Drop* routine first sets down the FTP so that the robot can acquire the retainer plate, and create the delivery configuration. For our example, the robot places the FTP on the CP's A surface. Sensors are used to ensure that the plate is properly delivered to the CP surface.

(7) The *Release Arm* routine liberates the arm from the FTP.

Arm Alignment to CP Base routine

Positioning of the robot's electromagnetic arm to the FTP uses the control algorithm shown in Figure. 4.27.

Algorithm Description:

(1) The controller positions the system X, Y, θ and ϕ axes above the processing configuration. The robot's electromagnetic arm is then lowered towards the FTP using the Z-axis's carriage. Ten millimetres above the FTP, movement is halted to allow the system to verify that the plate is actually present in the processing assembly. This also acts as a safety measure should the robot's arm be in the incorrect location.

(2) The controller uses the arm's magnetic proximity sensor to determine: (1) plate presence, and (2) arm alignment. A high sensor reading indicates that the arm is in the correct location and processing continues with the execution of step 3. A low sensor reading indicates that the plate is not present, or that the arm is not in the correct location. In this case, as a double check, the system moves the robot's arm 2 millimetres closer to the plate. If the sensor still does not detect the FTP, the controller sends an error message to the HMI and the Detach routine is aborted. Otherwise, processing continues with the execution of the next step.

(3) The arm is attached to the FTP.

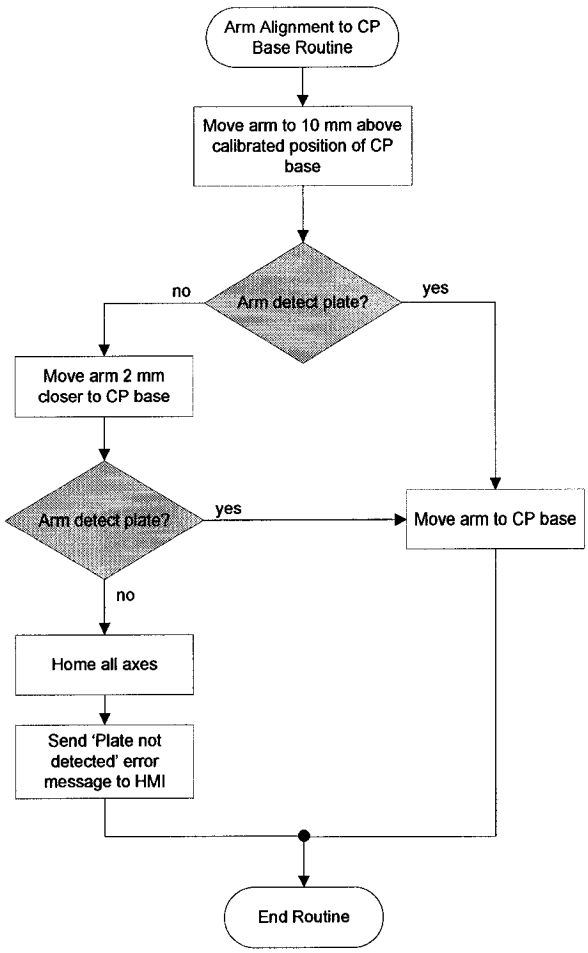


Figure 4.27: Control algorithm for the *Arm Alignment to CP Base* routine. The shaded bases represent the lower-order routines that the *Arm Alignment to CP Base* routine calls. The shaded decision boxes represent sensor readings.

Decouple Plates routine

Decoupling of the FTP from the filter plate is achieved by sending a polarity reversal pulse to the FTP's four electromagnets. In theory, one pulse should be sufficient to detach the plates from each other. However, in practice, this is inadequate as the pulse duration delivered is too short, and the plates reattach themselves before the motion controller can remove the FTP. Although it would be easier if the hardware could accommodate a longer pulse width, this option was not available because the current circuit design was fixed and could not be manipulated. Consequently, a software solution was found. Several approaches to decoupling, using different techniques from attempting to decouple in mid-air while the arm was being raised on the Z-axis, to adjusting the timing and actuation sequence of the various I/O channels were not

successful. Reliable decoupling was finally achieved by implementing the algorithm shown in Figure 4.28.

Algorithm Description:

(1) The controller pulses the FTP's electromagnets on and off while the Z-axis raises the arm in half millimetre increments causing the FTP to slowly disengage from the filter plate.

(2) Pulsing ceases when the Z-axis reaches a predefined location.

(3) The arm's proximity sensors are monitored to determine whether plate removal has occurred. If the sensor is active, FTP decoupling was successful. If not, the arm will make four more attempts to decouple the FTP, before it concedes defeat and sends an error message to the HMI application.

Plate Drop routine

Algorithm Description:

(1) The motion controller checks the status of the CP's A surface magnetic sensor to ensure that the surface is able to receive the FTP. If it determines that another plate already occupies the surface, the controller sends an error message to the HMI application, and waits for the operator to decide how to proceed. Two options exist: (1) remove the extraneous plate and resume processing, or (2) discontinue processing of this step.

(2) The robot moves towards the CP surface and proceeds to gently place the FTP onto its surface.

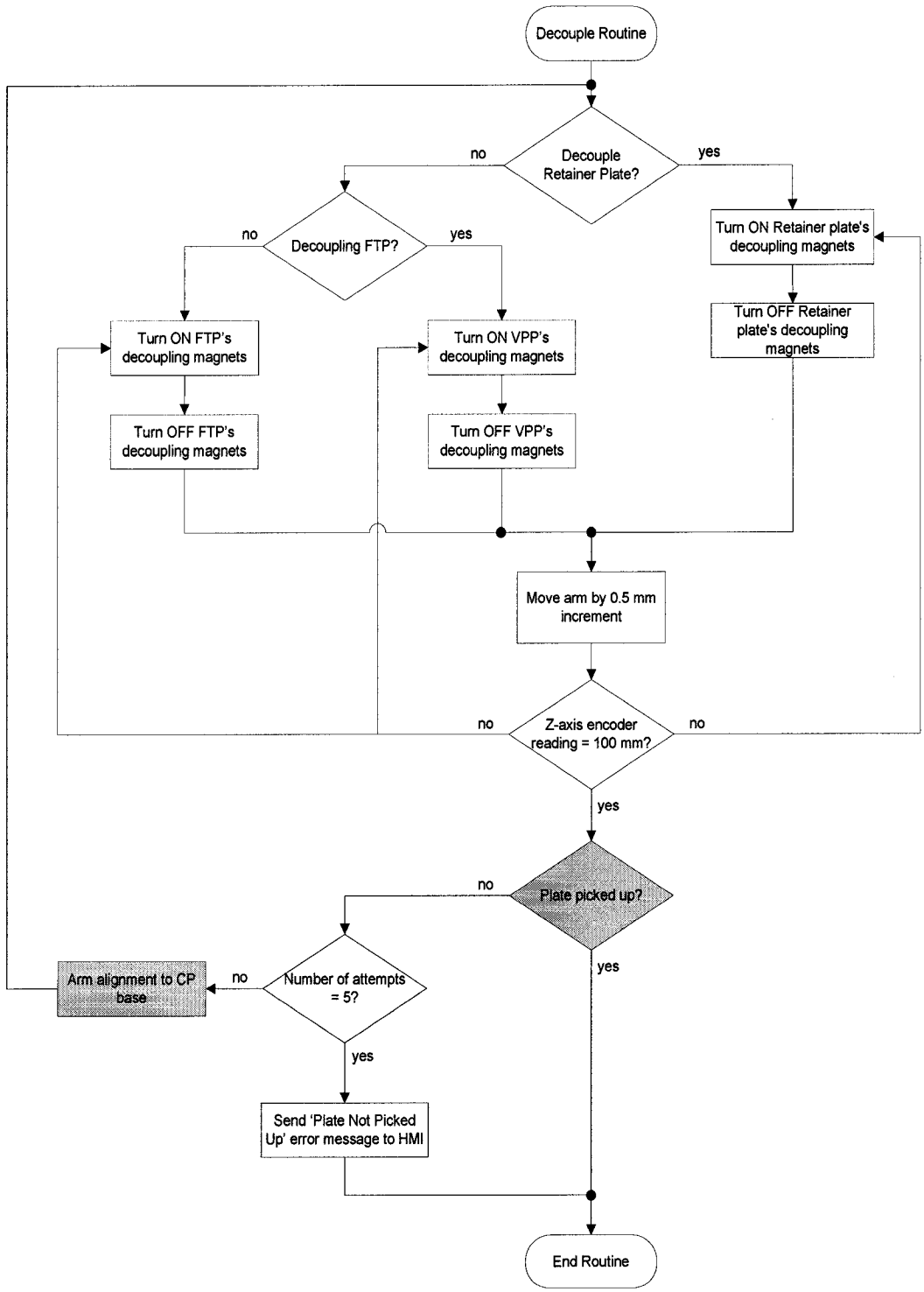


Figure 4.28: Control algorithm for *Decouple* routine. The shaded base represents the lower-order routine that the *Decouple* routine calls. The shaded decision box represents a sensor reading.

Release Arm routine

Arm decoupling is straightforward and involves four steps.

Algorithm Description:

(1) The controller turns off the arm's four H-bridge electromagnets.

(2) The Z-axis, and thus the arm, is moved to the Z-axis home sensor.

(3) The controller measures the signal from the arm's proximity sensor, to ensure that the arm is no longer attached to the plate. In this case, a low reading indicates that arm decoupling was successful and the controller can then proceed to execute step 4. Conversely, if the reading is high, the Z-axis returns to the CP surface and the system re-executes steps 1 and 2. If after five attempts, the arm is still unable to detach itself from the FTP, the controller signals an error to the HMI application, waits for the operator to manually remove the FTP from the arm, and terminates the routine.

(4) All axes are sent home.

Attach Plates routine

Although the aims of the *Attach Plates* and *Detach Plates* routines are opposite, the first is to join two plates while the second it is to separate them; their control algorithms and the routines they call are very similar. Figure 4.29 shows the sequence of events used to attach the retainer plate onto the filter plate/VPP assembly.

Algorithm Description:

(1) The controller determines whether the retainer plate is located on the CP's B surface by probing the sensor embedded within the surface. If the retainer plate is in the correct location, the sensor will give a digital high reading; the robot will then actuate the arm's electromagnets and move towards the retainer plate. Otherwise, the controller will send an error message to the HMI application, wait

for the operator to place the plate on the CP and either continue processing, or terminate processing altogether.

(2) The *Plate Pickup* routine is called to position the arm to the CP's B surface and pickup the retainer plate. As for the *Arm Alignment to CP Base* routine, this routine moves the axes to 20 millimetres above the retainer plate and then halts all motion. The robot then confirms, using the arm's magnetic proximity sensor, whether it is in the correct position to pickup the retainer plate. If the error in alignment between the arm and retainer plate is unacceptable, the motion controller will send an error message to the HMI application advising the operator to recalibrate the arm for this location.

(3) The Z-axis lifts the arm 20 millimetres above the CP's B surface, and the motion controller verifies if the plate is present on the arm. If the controller does not detect the plate, the arm will reattempt to pickup the retainer plate up to five times, before sending an error message to the HMI and terminating the processing of this procedure. Otherwise, it will proceed to step 4.

(4) The *Retract CP – OPEN* routine is used to retract the CP.

(5) The *Arm Alignment to CP Base* routine drives the arm to gently place the retainer plate onto the 2-plate configuration, located on the CP's base.

(6) The arm detaches from the retainer plate by deactivation of the arm's electromagnets (*Release Arm*).

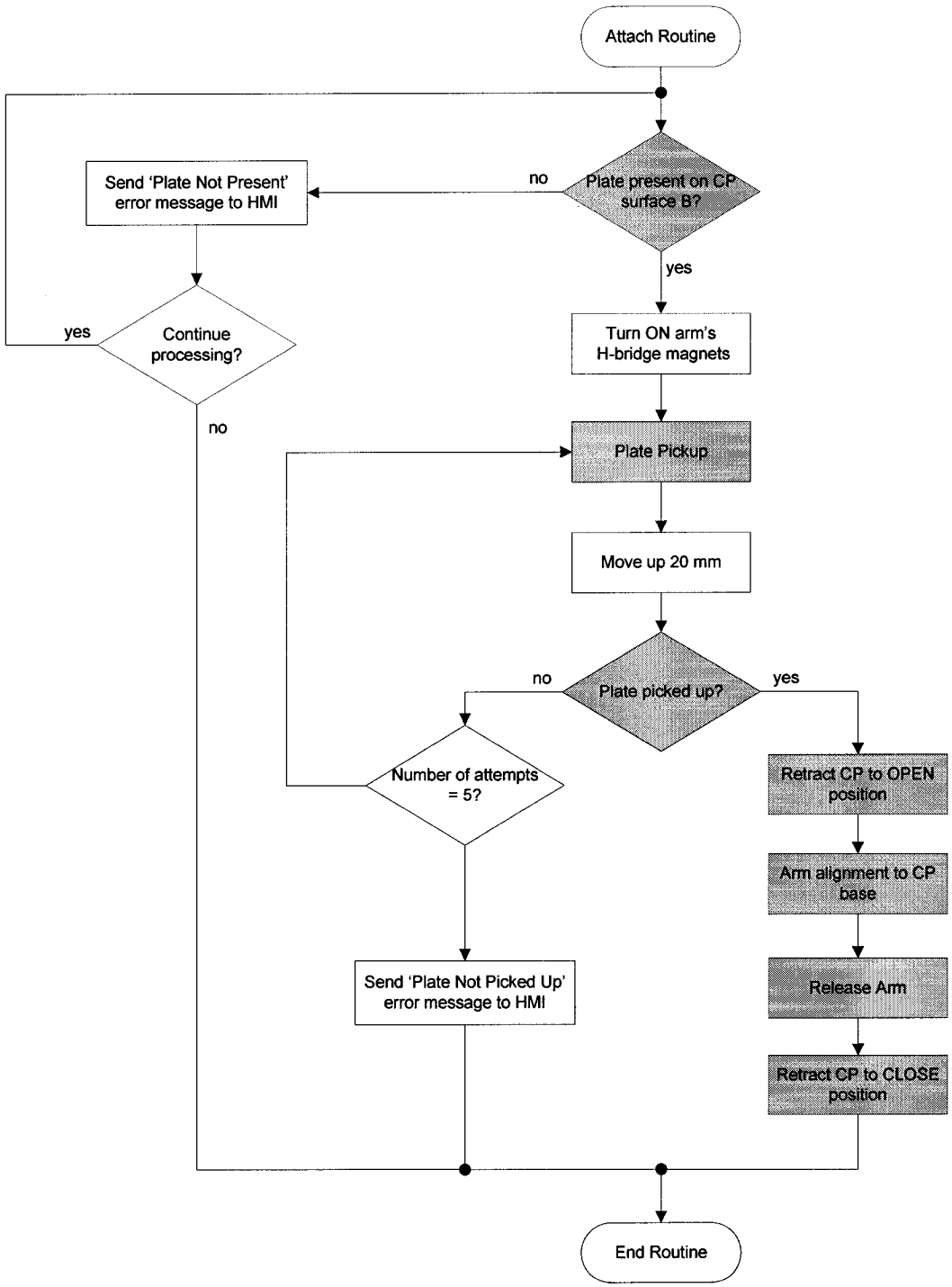


Figure 4.29: Control algorithm for the *Attach Plates* routine. The shaded bases represent the lower-order routine that are called by the *Attach Plates* routine. The shaded decision boxes represent sensor readings.

5 Software Validation

5.1 Chapter Overview

The reader should have a solid understanding of the hardware and software implemented to achieve an autonomous, high-throughput, automated sample preparation system. This chapter will present data demonstrating that the EMPOP robot operates as expected. It will be divided into five sections, corresponding to the system's main processes, and will use log files generated by the robot to show that each process is capable of repeatable function with different operational parameters. The chapter concludes by illustrating results from a complete sample preparation run, in which the robot prepared 24 samples simultaneously in an automated manner.

5.2 Software Validation of System Processes

Validation of the EMPOP system's five main processes: (1) plate cooling, (2) sample filtration, (3) sample ejection, (4) fluid handling and (5) plate manipulation, was achieved by executing each process multiple times, to verify whether it operated as programmed and designed. The duration of each process was compared over several iterations to determine if it remained constant; as a constant duration proves that the process is capable of repeatable operation. In addition, the sample filtration, sample ejection and fluid handling processes were tested under different initial conditions to ensure that they operated in a similar manner and that each trial followed the same trend.

Plate Cooling

The operator may initiate the cooling sequence at any time during a run. Figure 5.1 shows the temperature of the cooling platform for three different trials. The data for each trial was obtained from the cooling file. From the figure, it is evident that the cooling mechanism is able to cool the platform to ~ 4°C. Referring to Table 5.1, we can conclude that as expected the time required for the platform to reach 4°C depends on the ambient temperature in the room prior to initiating cooling. Lower ambient temperatures

result in quicker cooling times, while higher ones place more stress on the cooling mechanism making cooling more challenging. It is important to note that the data throughout this chapter is represented to the same precision as in the log file, 1ms. Since total times are in the order of seconds, for easier reading and comprehension, times were converted from milliseconds to seconds.

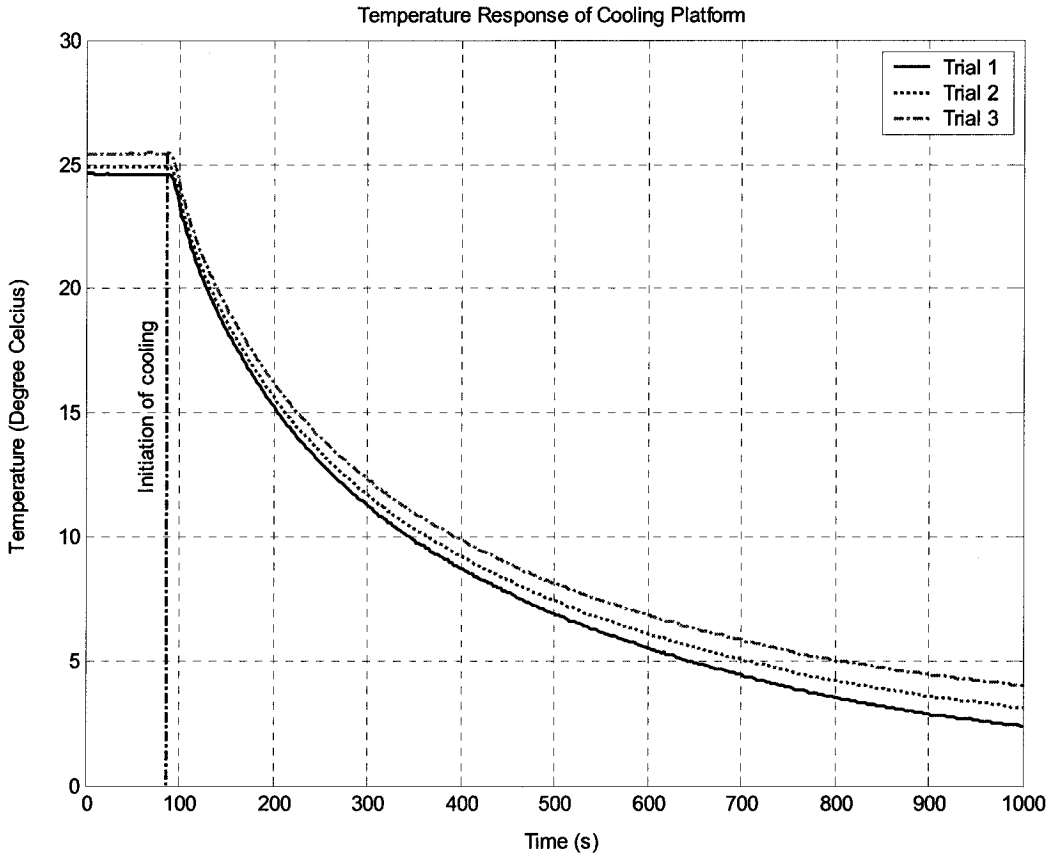


Figure 5.1: Temperature response of cooling platform

Process	Initial Temperature	Time to reach 4°C
Trial 1	24.65°C	750 s
Trial 2	24.95°C	835 s
Trial 3	26.11°C	1000 s

Table 5.1: Effect of ambient temperature on rate of cooling

Sample Filtration

Three experiments were devised and conducted to asses the sample filtration process's operation.

Experiment A

The first experiment was designed to verify whether the filtration algorithm actuated the vacuum release solenoid valves for 10 seconds, as programmed, irrespective of the reservoir's initial vacuum level. Table 5.2 shows the data collected from the main log file for two trials of the sample filtration process; each trial was executed using different filtration levels and consisted of three repetitions. Since both trials have similar mean and standard deviation (std.), we can conclude that the system consistently actuated the valves for 10 seconds regardless of the level of vacuum generated and the number of repetitions.

Trial	Initial Transducer Level	Rep. 1	Rep. 2	Rep. 3	Mean	Std.
Trial 1	4V	10.012s	10.012s	10.009 s	10.011s	0.001s
Trial 2	3V	10.010s	10.009s	10.013 s	10.010s	0.002s

Table 5.2: Sample filtration data to determine valve actuation duration

Experiment B

The second experiment was designed to determine whether the sample filtration's first-order control algorithm was capable of repeatable execution. The total time required for the entire sample filtration process, to empty the vacuum reservoir and filter the samples for a single repetition, was measured for three trials. The timing data for each trial is shown in Table 5.3. With a mean duration of 28.525 ± 0.489 seconds, the algorithm is clearly repeatable over several iterations.

Initial Transducer Level	Trial 1	Trial 2	Trial 3	Mean	Std.
4V	28.749 s	27.954 s	28.872 s	28.525s	0.498 s

Table 5.3: Sample filtration data to determine process repeatability

Experiment C

The aim of the third experiment was to determine whether the filtration process actually delivers vacuum to the VPP. To test this, the sample filtration process was conducted three times using different levels of vacuum corresponding to a vacuum transducer voltage of 4V, 3.8V and 3.6V, respectively. After each trial, the vacuum cylinder was opened to the atmosphere to ensure that no residual vacuum remained in the tank. The system's response was determined by capturing vacuum transducer readings for each trial, over a 10 second period after the filtration release valves were actuated. From Figure 5.2, we can see that the vacuum level decreases at a constant rate for all three trials. Moreover, the vacuum level is held constant in all three cases and only begins to fall at the point the valve is actuated. Therefore, the results confirm that the process functions as expected, filtering the samples at a constant rate, and actuating the valve at the correct time, irrespective of the initial vacuum level.

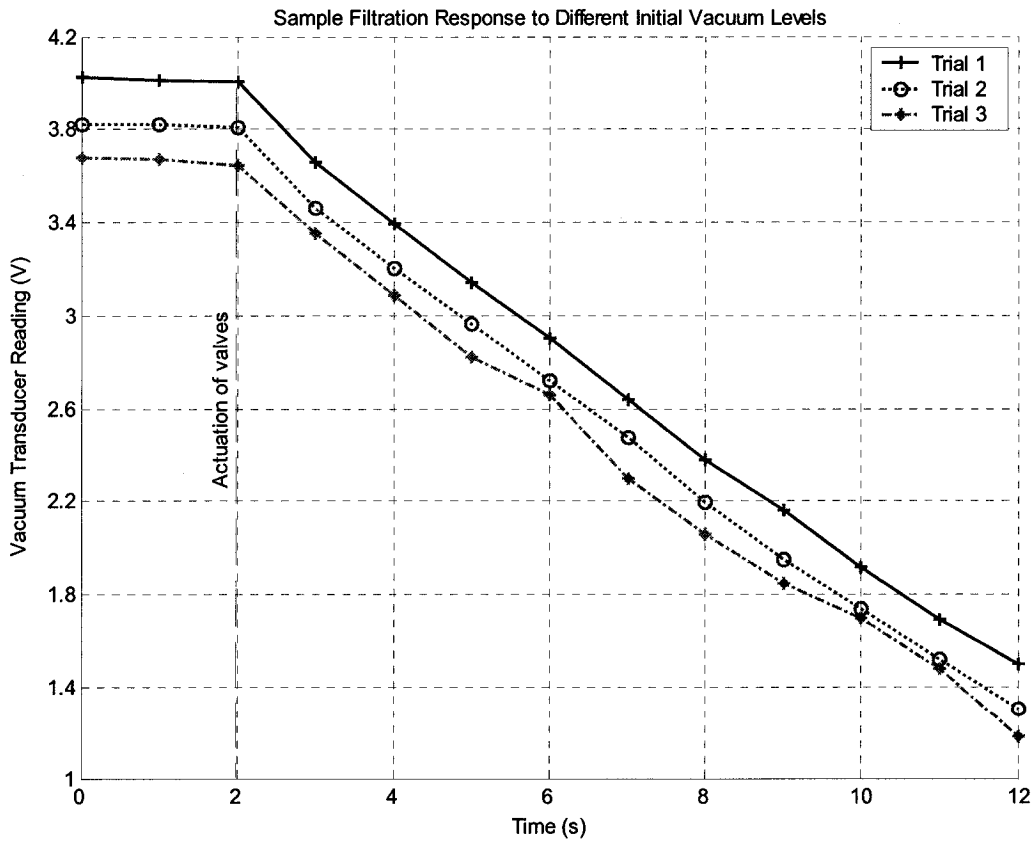


Figure 5.2: Sample filtration response to different initial vacuum levels

Sample Ejection

As explained in the previous chapter, sample ejection is similar to sample filtration except that the actuation duration for the pressure release valve is not fixed. Rather, the operator dictates how the valves are actuated for each repetition by specifying a pulse sequence, defined by the number of pulse cycles and the active pulse duration (APD). Figure 5.3, depicts the two pulse sequences, where each pulse segment is labelled as APD 1, APD 2, APD 3 and APD 4, respectively. Moreover, a high pulse segment indicates that the valve is open

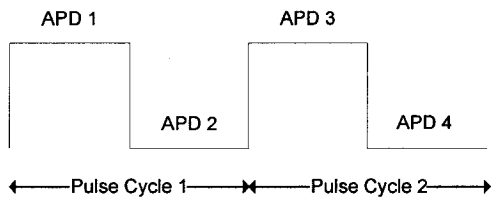


Figure 5.3: Pulse sequence used

Three experiments were carried out to validate the sample ejection process.

Experiment A

The objective of the first experiment was to determine how well the sample ejection algorithm generates different pulse sequences. This was accomplished by comparing the APD for two sample ejection trials; each consisting of a single repetition and using a pulse sequence with two pulse cycles. Trial 1 used an APD of 1 second, whereas trial 2 used an APD of 1.5 seconds.

Trial	Specified APD	APD 1	APD 2	APD 3	APD 4	Mean	Std.
Trial 1	1 s	1.066 s	1.083 s	1.073 s	1.101 s	1.080 s	0.015 s
Trial 2	1.5 s	1.565 s	1.589 s	1.561 s	1.582 s	1.574 s	0.013 s

Table 5.4: Sample ejection data to determine APD

Table 5.4 summarizes the results of the two trials. The mean APD for first trial was 1.080 ± 0.015 seconds, whereas the mean for the second trial was 1.574 ± 0.013 seconds. Moreover, Table 5.5 shows that the duration of both pulse cycles were similar for each

trial. The difference of ~ 1 second between the duration of the pulse cycles in trial 1 and those in trial 2, reflects the difference in APD between the two trials. Thus, the system generates the specified APD and number of pulse cycles.

Trial	Duration of Pulse Cycle 1	Duration of Pulse Cycle 2	Mean	Std.
Trial 1	2.149 s	2.174 s	2.161 s	0.017 s
Trial 2	3.154 s	3.143 s	3.148 s	0.007 s

Table 5.5: Sample ejection data to measure the duration of each pulse cycle

Experiment B

The second experiment was designed to determine the algorithm's ability to reproduce sample ejection. This required gathering timing information for three trials of the sample ejection process. Each trial used the same pressure level, included one repetition, and used a pulse sequence consisting of three pulse cycles with an APD of 1 second. It is important to note that filling the pressure reservoir air requires considerably more time than creating vacuum. Consequently, these trials took significantly longer than those for sample filtration. The data obtained for this test (Table 5.6) shows that sample ejection algorithm is repeatable over several trials.

Initial Transducer Level	Trial 1	Trial 2	Trial 3	Mean	Std.
3V	117.555 s	117.824 s	114.686 s	116.688 s	1.739 s

Table 5.6: Sample ejection data to determine process repeatability

Experiment C

The last experiment was used to verify whether the robot could deliver various levels of pressurized air to the VPP. It involved recording the pressure transducer readings for three trials of the sample ejection process, in which the pressurization level was varied to match pressure transducer levels of 4V, 3.8V and 3.6V, respectively. Each trial was executed using the same operating conditions as in Experiment B. The results for the three trials are shown in Figure 5.4; clearly, the algorithm does indeed release the pressure in three short bursts. Moreover, the pressure release is consistent for each trial regardless of pressurization. As expected, the volume of pressure in the reservoir decreases each time the valve is opened (APD1, APD3 and APD5), whereas it plateaus

when the valve is closed (APD2, APD4 and APD5). This last experiment illustrates that the EMPOP system is able to consistently perform sample ejection for any level of pressurization and release pattern.

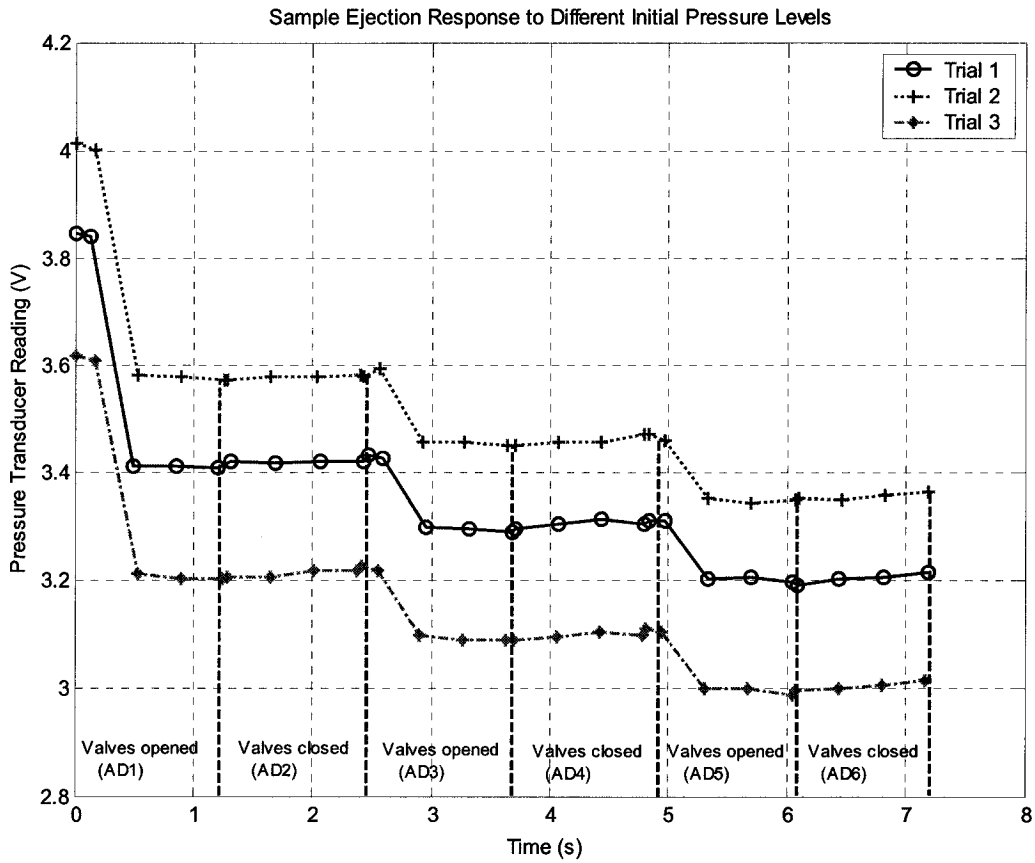


Figure 5.4: Sample ejection response to different initial pressure levels

Fluid Handling

Three experiments were performed to validate the fluid handling process.

Experiment A

The goal of the first experiment was to assess the first-order algorithm's ability to dispense and aspirate reagent to and from the core mechanism. The duration for two trials of the fluid handling sequences, including priming, reagent dispensing, incubating and reagent aspiration, were measured. The volume, number of repetitions, incubation time and the number of wells to process were kept fixed.

Table 5.7 shows the timing data for the four lower-order routines that are called by the *Fluid Handling* routine. The total time is the sum of the durations for each lower-order routine. The mean total time for the two trials is 737.916 ± 1.528 seconds. Since the deviation between trials is a marginal one fifth of one percent, this illustrates that the first-order routine and the lower-order routines it calls operate in a repeatable manner.

Trial	*Prime Lines and Needles*	*Dispense Reagent*	*Incubation*	*Aspirate Reagent*	Total
Trial 1	150.002 s	228.438 s	59.907 s	300.650 s	738.997 s
Trial 2	148.991 s	227.928 s	60.013 s	299.901 s	736.836 s

Table 5.7: Fluid handling data to determine process repeatability

Experiment B

This experiment was designed to determine how well the process responded to parameter modifications. The duration for two trials of the fluid handling process were measured. Apart from the incubation time, which was doubled from 1 minute in trial 1, to 2 minutes in trial 2, all other parameters were kept constant. The incubation time parameter was chosen to be varied because its effect on the system's response is predictable. For example, if all parameters except the incubation time are constant, then the total time between two trials should differ by the difference in incubation time. This does not hold for any of the other parameters, as their behaviour with respect to time is not easily known.

The timing data for the two trials, extracted from the system's main log file, is shown in Table 5.8. The incubation time measured for trial 1 was 60.009 seconds (1.000 minutes), whereas for trial 2 it was 120.008 seconds (2.000 minutes). These values correspond to the incubation times specified. As expected, the timing data for priming, dispensing and aspiration was the same, as all the other operational parameters were identical for both trials. In addition, the difference in the overall duration of the two trials was 62 seconds (~1.033 minutes), which corresponds to the second trial's additional minute of incubation. These data clearly validates the algorithm's ability to function under varying initial conditions.

Trial	*Prime Lines and Needles*	*Dispense Reagent*	*Incubation*	*Aspirate Reagent*	Total
Trial 1	150.162 s	228.080 s	60.009 s	301.124 s	739.375 s
Trial 2	150.334 s	228.032 s	120.008 s	299.120 s	797.494 s

Table 5.8: Fluid handling response to different incubation times

Experiment C

The aim of this experiment was to determine whether the process was actually able to manipulate fluids. It consisted of dispensing three different volumes of water to each of the system's four needles. Additionally, in order to demonstrate repeatability of dispensing fluids, the experiment was conducted twice for each volume dispensed. Although a simple experiment, it tested the robot's ability to control the system's network of fluid valves and pumps, as well as prime the system's fluid lines and dispense the desired volume of reagent. The volume was determined by delivering water to 12 eppendorf tubes, whose weight were measured before and after reagent dispensing. Since the density of water is 1.0 mg/ml, the difference in weights was sufficient to determine the volume dispensed.

Volume	Needle A	Needle B	Needle C	Needle D	Mean	Std.
300	303.100 µl	306.871 µl	305.282 µl	306.038 µl	305.322 µl	1.617 µl
300	302.761 µl	306.109 µl	305.966 µl	304.473 µl	304.827 µl	1.563 µl
500	506.279 µl	509.656 µl	507.403 µl	507.849 µl	507.796 µl	1.404 µl
500	505.783 µl	507.821 µl	508.124 µl	506.936 µl	507.166 µl	1.050 µl
800	805.763 µl	808.075 µl	808.623 µl	807.120 µl	807.395 µl	1.252 µl
800	805.932 µl	809.714 µl	807.998 µl	806.950 µl	807.648 µl	1.614 µl

Table 5.9: Fluid handling data to illustrate process function

Table 5.9 summarizes the data obtained after weighting all eppendorf tubes. The data clearly shows that the process is able to deliver the required reagent repeatability and in a standardized manner. Moreover, it proves that the hardware setup and software routines are capable of delivering fluid to the core mechanism using all four needles.

Plate Manipulation

The plate manipulation process does not involve any operator specified parameters. Consequently, determining the process's performance and repeatability is much simpler, and involved executing four trials of the plate decoupling and placing steps for each plate they affect. To ensure comparable results the axes were homed to their zero reference locations prior to starting each iteration.

The data for the four trials of the *Attach Plates* and *Detach Plates* routines are summarized in Table 5.10. The mean time for placing the VPP and the retainer plate were very similar as expected because the start and end locations for both plates were identical. Following the same rationale, we expect the mean time to decouple the FTP and retainer plate to be similar, since the distance travelled by both plates to reach the cooling platform's A surface is the same. The slightly longer decoupling time for the VPP arises because the robot has to move ~10 mm further to move the VPP to the cooling platform's B surface than to it's A surface.

Attach Plates	Trial 1	Trial 2	Trial 3	Trial 4	Mean	Std.
VPP	259.621 s	261.112 s	258.386 s	260.402 s	259.880 s	1.167 s
Retainer	260.134 s	262.908 s	262.101 s	261.998 s	261.785 s	1.173 s
Detach Plates	Trial 1	Trial 2	Trial 3	Trial 4	Mean	Std.
FTP	472.415 s	473.032 s	472.293 s	471.288 s	472.257 s	0.722 s
Retainer	471.120 s	471.658 s	470.909 s	469.191 s	470.719 s	1.066 s
VPP	475.617 s	478.903 s	477.278 s	476.991 s	477.197 s	1.348 s

Table 5.10: Plate manipulation data to determine process repeatability

5.3 Software validation of the automated protocol

The last section explained how each of the robot's five main processes were validated to determine whether they operated as expected. This last experiment was conducted to determine how well the EMPOP robot functioned as a whole and to verify whether it was actually capable of executing an entire automated protocol.

Figure 5.5 is a photograph of 24 samples prepared using the EMPOP system. The samples are located in the center of the photograph in rows 5 through 8. Although, the consistency of the prepared samples (dark spots) as you move across the rows (from A to H) is not uniform, the samples are present, and so prove that the robot's two-tier software package functions to render a fully automated system. The samples were prepared following protocol 1, where each step was executed in manual mode. The only step not executed by the robot was plate inversion, as the transfer platform was not operational when the experiment was conducted.

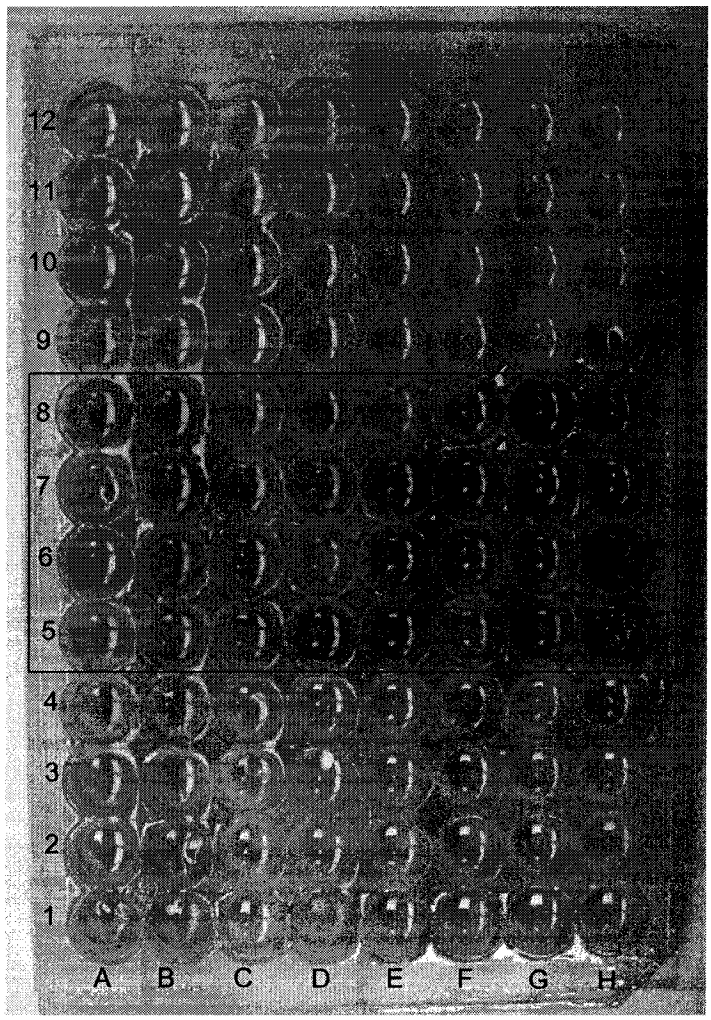


Figure 5.5: Photograph of 24 samples prepared using the EMPOP robot

At this stage in the development of the prototype, improving sample quality requires optimizing the system's operational parameters to determine the ideal set-values for sample filtration, sample ejection and fluid handling. As mentioned in the previous

chapter, the software enables the operator to test individual steps to determine the samples response to varying parameter levels. The status of each step is also updated automatically, enabling the operator to keep track of the steps that were completed successfully. Furthermore, the software enables the operator to halt and re-start processing in the event of unexpected behaviour such as a needle crash.

These features facilitate the modification of system parameters to improve sample yield and quality. The software provides all the functionality required for an experienced operator to optimize EMPOP operation. Moreover, the incorporation of the log file gives the operator a tool for quantitatively verifying the system's behaviour and function over several runs.

6 Conclusion

6.1 Chapter Overview

This chapter will summarize my contribution towards the development of the EMPOP system. The chapter begins with an outline of the thesis objectives and a brief discussion of the software requirements necessary for a functioning prototype. It continues with a summary of the software architecture that was implemented and justifies its use. Finally, the chapter concludes by pinpointing areas that require further attention and possible methods for resolving them.

6.2 Summary of Thesis Work

Like genetics in its day, proteomics over the past few years has slowly become a household name. The scale of the proteomics initiative is significantly larger than it was when researchers were first beginning to sequence the human genome. As each day passes, the quest to discover the function, structure, behaviour of and interaction between proteins is growing and changing. Consequently, there is overwhelming pressure on scientists, technicians and engineers from various technical backgrounds, to develop new instruments and methods to increase the throughput and decrease the wait times required to conduct experiments or gather results.

Within the realms of organellar proteomics, one of the major challenges faced by researchers is their inability to obtain a homogenous starting sample. As a result, researchers must present their samples to a transmission electron microscope before and after MS analysis to confirm sample purity and localize the isolate protein[1,10,25]. Current TEM sample preparation methods are extremely time- consuming and labour-intensive. To overcome these problems, the EMPOP robot was designed and developed to completely automate and standardize the preparation of up to 96 subcellular sample fractions.

This thesis describes the software methodology and architecture used to control the EMPOP system. To transform the mechanical system into a functioning robotic entity, the software had to meet four objectives:

(1) To eliminate the operator's need to learn MP and have knowledge of its commands.

(2) To create a customisable, visual, executable application which would permit the operator to control and modify robot function.

(3) To organize system parameters and store data so the operator can easily repeat experiments and compare the robot's performance over several runs.

(4) To establish the firmware routines that will enable the robot to carry out all aspects of the automated sample preparation protocol.

To attain these goals, I had to select the software framework that would meet these objectives. Choosing the right software architecture proved to be complicated as many potential software structures and implementations were available. However, since the system's hardware is structured following a PC-based automation scheme, I decided to use a client-server software architecture, as it mimicked and complimented the two-tier nature of the hardware architecture. More importantly, the development of this two-tier hardware/software architecture ensured separation between the operator and the control unit, and so helped achieve the first objective.

The second objective was achieved via the operator echelon (PC and client software) of the two-tier architecture, which was developed to provide the operator with a "friendly and flexible" means of interacting with the robot. The HMI application was programmed using VB and was divided into six GUI's: (1) Ethernet connection, (2) Calibration Setup, (3) System Setup, (4) Reagent Setup, (5) Process Control and (6) Process Monitoring. In addition, to satisfy the third objective, a configuration file and log file

were built-in to the HMI to enable the operator to retrieve parameter settings from past runs and save data from a current run.

Conversely, the control echelon (robot and server software) is isolated from the operator. It consists of low-level routines that interact with the HMI to control the robot. These routines were developed using MP, and were separated into different top-down structures depending on which GUI they were associated with. In addition, they were assigned to different layers depending on their complexity and function. The first-order control routines associated with the Process Monitoring GUI are responsible for controlling and governing the system's five main processes: (1) plate cooling, (2) sample filtration, (3) sample ejection, (4) fluid handling, and (5) plate manipulation. Great care was taken to ensure a modular design, and clearly define the firmware routines in an effort to satisfy the fourth objective.

All EMPOP processes were tested rigorously and independently under varying initial conditions to confirm their functionality and assess their robustness. Integration of all the system's subcomponents was tested using the robot to prepare 24 subcellular samples in a totally automated manner. Although improvements still need to be made to optimize sample preparation, this manuscript has shown the software to be robust, repeatable and operational. Consequently, it has proven itself to be an invaluable tool to the operator.

6.3 Future Work

This thesis has shown that the software architecture implemented meets the thesis objectives, however there are three areas within the software that could be improved.

The first improvement is with regard to the system error-handling mechanism. Currently, errors generated by the system cause processing to halt and the robot to move gracefully towards a predefined stow location; the operator must then continue processing manually. In the future, a more robust error-handling algorithm is required to

create a fully autonomous robotic system. Improving the system's error-handling capabilities requires testing the system under perturbed and strained conditions to discover system flaws and weaknesses. Moreover, it would require working with the operator to determine the ideal hardware response in the case of a malfunction.

The second improvement would involve implementing a software scheme to enable the operator to completely customize the sample preparation protocol. Currently, the operator can select between one of two protocols, using a fixed reagent list. Ideally, the operator should be able to move, add or remove steps to the current protocol as well as use any reagents they wish. Achieving this would require a more flexible Setup GUI. Moreover, it would entail using more MP communication variables to enable the HMI application to transmit modifications to the motion controller. Since the number of variables is fixed, this would require the programmer to further optimize variable usage. The implementation of this improvement, with the addition of a generic barcode reader, would enable the robot to determine the ideal reagent list for a sample based on its SOP-barcode. This would further ensure that samples were processed in a standardized, repeatable manner.

The third improvement is to create a System Testing GUI, to allow the operator to test any component or I/O channel within the system. Currently, aside from actually running a step, the software does not provide the operator with a means of debugging the system when a hardware malfunction or failure occurs. Once implemented, this additional GUI would enable the operator to ascertain the status of a hardware component by simply clicking on a button, which would transparently translate into a MP command to actuate an I/O channel or control algorithm.

References

[1] Alberts, Bruce, *et al.* "Chapter 9: Visualizing Cells." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 547-583.

[2] "Cell Biology." Wikipedia: The Free Encyclopedia. 2005. Wikipedia. 13 October 2005 < http://en.wikipedia.org/wiki/Cell_biology>

[3] Cooper, Geoffrey M. "Eukaryotic Cells." The Cell: A Molecular Approach. 2nd ed. Sunderland (MA): Sinauer Associates, Inc., 2000. 9-12.

[4] Alberts, Bruce, *et al.* "Chapter 2: Cell Chemistry and Biosynthesis." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 47-129.

[5] Alberts, Bruce, *et al.* "Chapter 12: Intracellular Compartments and Protein Sorting." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 659-711.

[6] Campbell, Neil. A, *et al.* Biology. 6th ed. Benjamin Cummings, 2001.

[7] Alberts, Bruce, *et al.* "Chapter 1: Cells and Genomes." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 3-47.

[8] Alberts, Bruce, *et al.* "Chapter 6: How Cells Read the Genome: From DNA to Protein." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 299-375.

[9] Donaldson, Logan. "Biology 1010." DonaldsonLab. 2005. York University: Department of Biology. 13 October 2005 <http://dryden.biol.yorku.ca/1010/sheets/FunctionsOfProteins.pdf>

[10] Alberts, Bruce, *et al.* "Chapter 3: Proteins." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 129-191.

[11] "About the Human Genome Project." Human Genome Project Information. 2004. U.S. Department of Energy Office of Science. 13 October 2005 <http://www.ornl.gov/sci/techresources/Human_Genome/project/about.shtml>

[12] "Proteomics Introduction." Bioinformatics Web: Comprehensive Educational Resource on Bioinformatics. 2005. 13 October 2005 <http://www.geocities.com/bioinformaticsweb/proteomicsintro.html>

[13] Proteomics Explained. 2004. Proteome Systems Limited. 13 October 2005. <http://www.proteomesystems.com/home/profile.asp?MenuID=37&RefMenuID=21&TopMenuID=26&LeftMenuID=37>

[14] Mann, Mathias. "Organellar Proteomics: Providing a Cellular Home for Every Protein." The Scientist 18.7 (2004): 32-34

[15] Hester, Doyle A., *et al.* "Posttranslational Modifications of Self-Antigens." Annals of the New York Academy of Sciences 1050 (2005): 1-9

[16] Taylor, Steven W., *et al.* "Global Organellar Proteomics." Trends in Biotechnology 21.2 (2003): 82-88

[17] Aebersold, Ruedi, *et al.* "Mass Spectrometry-Based Proteomics." Nature 422 (2003): 198-207

[18] Tyers, Mike, *et al.* "From Genomics to Proteomics." Nature 422 (2003): 193-197

[19] Jung, Eva, *et al.* "Proteomics Meets Cell Biology: The Establishment of Subcellular Proteomes." Electrophoresis 21 (2000): 3369-3377

[20] "TEM Class Notes." Microscopy Courses. University of Georgia: Center for Ultrastructural Research. 13 October 2005 <http://www.uga.edu/caur/temindex.htm>

[21] Bell, Alexander W., *et al.* "Proteomics Characterization of Abundant Golgi Membrane Proteins." The Journal of Biological Chemistry 276.7 (2000): 5152-5165

[22] Steen, Hanno, *et al.* "The ABC's (and XYZ's) of Peptide Sequencing." Nature 5 (2004): 699 - 711

[23] Alberts, Bruce, *et al.* "Chapter 8: Manipulating Proteins, DNA, and RNA." Molecular Biology of the Cell. 4th ed. New York: Garland Publishing, 2002. 469-547.

[24] Padh, Harish. "Organelle Isolation and Marker Enzyme Assay." Tested Studies for Laboratory Teaching 13 (1992): 129-146

[25] Lodish, Harvey, *et al.* "Purification of Cells and Their Parts." Molecular Cell Biology. 5th ed. New York: W. H. Freeman & Co. 2003. 178-184

[26] Lodish, Harvey, *et al.* "Hierarchical Structure of Proteins." Molecular Cell Biology. 5th ed. New York: W. H. Freeman & Co. 2003. 60-68

[27] "Helix Systems Scientific Databases." Computational Molecular Biology at NIH: Databases. 2005. NIH. 13 October 2005 <http://molbio.info.nih.gov/helixdb.php>

[28] Huber, Lukas A., *et al.* "Organellar Proteomics: Implications for Subcellular Fractionation in Proteomics." American Heart Association. 92.9 (2003): 962-968

[29]"The Transmission Electron Microscope." Microscopes. 2005. NobelPrize.Org. 13 October 2005 <http://nobelprize.org/physics/educational/microscopes/tem/index.html>

[30] "Akashi 002B Transmission Electron Microscope." CEMMA Instruments. University of Southern California: Center for Electron Microscope and Microanalysis. 13 October 2005 <http://www.usc.edu/dept/CEMMA/_images/akashi-3%20web.jpg>

[31] Morrissey, David. Electron Microscopy. 2001. University of Bristol: Department of Biochemistry. 13 October 2005 <http://www.bch.bris.ac.uk/staff/pfdg/teaching/em.htm>

[32] Janeway, Charles A., *et al.* "Appendix I. Immunologists' Toolbox." Immunobiology. 5th ed. New York and London: Garland Publishing, 2001. 613-661.

[33] Lavoie, C. *et al.* "Roles for α2p24 and COPI in Endoplasmic Reticulum Cargo Exit Site Formation." The Journal of Cell Biology. 146.2 (1999): 285-299

[34] "Electron Microscopy." Wikipedia: The Free Encyclopedia. 2005. Wikipedia. 13 October 2005 <http://en.wikipedia.org/wiki/Electron_microscope>

[35] Harris, Tom. How Robots Work. HowStuffWorks. 13 October 2005 <http://electronics.howstuffworks.com/robot.htm>

[36] Encyclopedia: Industrial Robot. 2005. Natiomaster.com. 13 October 2005 <http://www.nationmaster.com/encyclopedia/Industrial-robot>

[37] 6K Controllers - Catalog (8000-4/USA). Parker Hannifin Corporation: Compumotor Division.

[38] The Visual Basic Programming Language. 1996. University of Michigan: College of Engineering and Computer Science. 13 October 2005 <http://www.engin.umd.umich.edu/CIS/course.des/cis400/vbasic/vbasic.html>

[39] 6K Series Command Reference. 1998. Parker Hannifin Corporation: Compumotor Division

[40] 6K Series Programmer's Guide. 1998. Parker Hannifin Corporation: Compumotor Division