Design of Control System for Quadruped Robot (4-Legged Robot)

by

YEE YUAN BIN

Dissertation submitted in partial fulfillment of
the requirements for the
Bachelor of Engineering (Hons)
(Mechanical Engineering)

MAY 2009

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Design of Control System for Quadruped Robot

(4-Legged Robot)

by

YEE YUAN BIN

A project dissertation submitted to the

Mechanical Engineering Programme

Universiti Teknologi PETRONAS

in partial fulfillment of the requirement for the

BACHELOR OF ENGINEERING (Hons)

(MECHANICAL ENGINEERING)

Approved by,

____

ROSMAWATI MAT ZAIN

Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

MAY 2009

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

YEE YUAN BIN

ABSTRACT

The project outcome is to design a control system for quadruped robot (4-legged robot). Early 2007, a quadruped robot was built by an UTP graduate; Mr. Tnay Chiat Siang (Matric No. 6114). The quadruped robot is well constructed in mechanical-wise; however, it has never been able to move as there is no motor driver and control system being developed for it. Therefore, the author has decided to come in on Mr. Tnay Chiat Siang's quadruped project and continue his work to develop a control system which will eventually drive the quadruped robot to perform crawling gait on flat and horizontal ground. The control system of quadruped involves gait control, stability control and motor control. This project is split to three phases. The timeline of each phase is that phase 1 is carried out at semester FYP 1 while phase 2 begins during the year-end semester break. Phase 3 is commenced during FYP 2. The work aspect of phase 1 is on schematic design and crawling gait planning of quadruped. A preliminary simulation is used to demonstrate the planned crawling gait. The focus of phase 2 is more on learning how to manipulate PIC microcontroller and servomotors. At last, phase 3 is the prototype fabricating and testing stage with the presence of servomotors and circuit board. At the end of project, the quadruped prototype is meant to perform forward crawling gait on flat and horizontal ground.

ACKNOWLEDGEMENT

First and foremost, I would like to thank immediate supervisor, Puan Rosmawati Mat Zain for her excellent guidance and advices. Her dedication of time and effort, relentless teaching and motivation has been the major factor towards the completion of my final year project. Besides, she has also been extremely supportive by giving me numerous of freedom and support in which it helps in unleashing my creativity and innovation.

Apart from that, I would like to express my thanks to the laboratory assistants Mr. Hafiz, Mr. Jani and Ms. Siti Fatimah for their supportive involvement in helping the project runs as scheduled.

Last but not least, I would also like to thank express my love and thanks to my family members for their understanding, support, love and encouragement throughout this period.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Background Study

Robot, a term that once represents an imagination and dream, has captured the interest of scientist for centuries. There are assorted types of robots today, either mobile or stationary robot, for instance, robot manipulator, wheeled robot, legged robot or etc. The idea of legged robot has come from Mother Nature; humanoid robot is a mimic of Homosapiens; biped (2-legged), quadruped (4-legged), hexapod (6-legged) and octopod (8-legged) robot are inspired by legged insects and mammals. The unique gait of legged insect and mammal such as climbing, crawling, jumping and etc. has inspired scientist to develop nature-like robot.

1.2 Problem Statement

Wheeled robot has less than satisfactory of maneuverability and movability on rugged terrain. Besides, vibration of wheeled robot moving on uneven surface has also cost the robot's lifespan and stability. Therefore, the idea of quadruped robot has come in to overcome such situation as it is able to maneuver on various surfaces.

1.3 Significance of Study

The outcome of this project will be an enhanced quadruped robot which is able to maneuver over various surfaces, such as even ground, rugged terrain, slope or even stairs. Thanks to the unique nature of quadruped robot, it is given the strength to carry out extreme duty such as bomb and landmine retrieval at war zone, search and rescue operation at earthquake-hit area, outer space and seabed exploration.

1.4 Objectives

The objective of this project is to:

- Design the control system to drive the quadruped to crawl on flat and horizontal ground.

1. 5 Scope of Work

The scopes of work include:

- Plan a suitable walking gait for quadruped robot in which the center of gravity (COG) will be constrained by its static stability margin. EXCEL spreadsheet will be used to create gait simulator.
- Study on choosing the suitable microcontroller and learn about programming microcontroller in C language as it provides higher flexibility in executing program instructions.
- Draw a CAD model and fabricate a prototype according to the CAD model.
- The quadruped equipped with motor controller will be tested on flat and horizontal ground to observe its mobility and maneuverability when performing crawling gait.

CHAPTER 2

LITERATURE REVIEW

2.1 Quadruped

Quadruped is a 4-legged robot which is inspired by the unique gait of insects and mammals, such as crawling, climbing, hopping and etc. Unlike the biped robot, it can stabilize its own body easily by adjusting its walking gait and body position. Therefore, due to its special nature of walking gait, it can be used for exploration in all terrain.

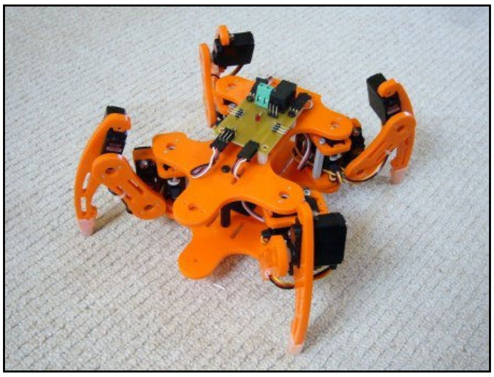


Figure 2.1: Example of Quadruped Robot

2.2 Leg Mechanism of 3 DOF

For a quadruped, a single leg of 3 degree of freedom leg mechanism is always favored due to its flexibility in performing various walking gait and the leg is usually divided into three sections, namely hip, thigh and lower thigh [1]. In such case, the number of degree of freedom of a quadruped with 3 degree of freedom of each leg will be 12. There are three joints for each leg: the first joint is laid between the body and the hip, the second joint is between the hip and the thigh while the third joint is between the thigh and the lower thigh [1].

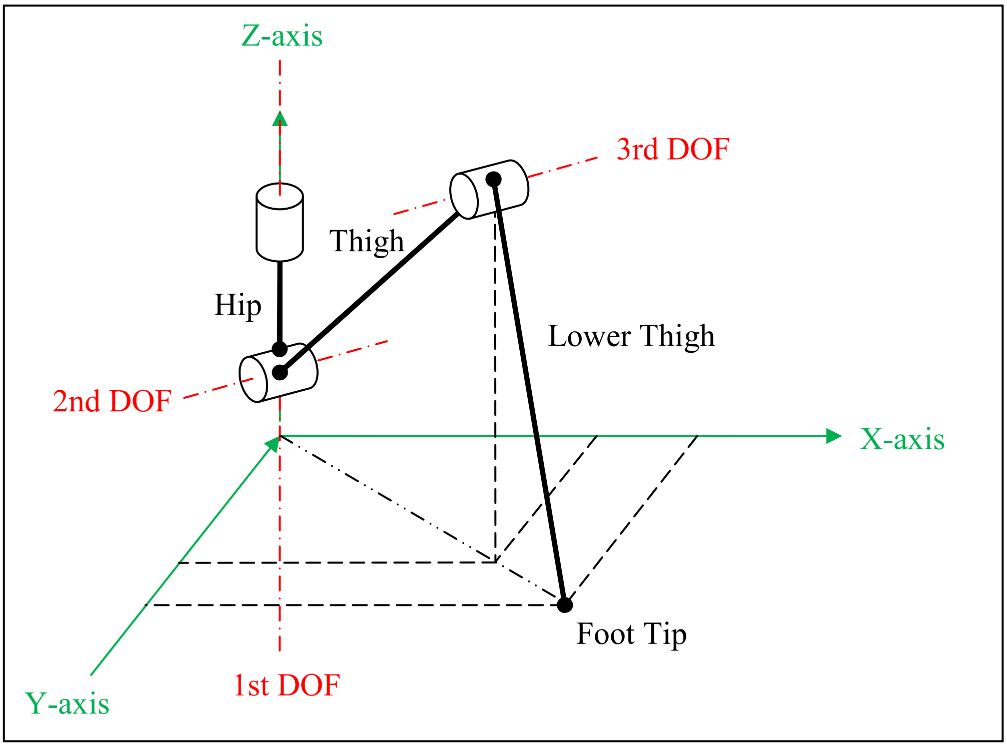


Figure 2.2: 3 DOF of Single Leg Mechanism

One joint representing 1 degree of freedom and the joint is usually driven by either actuator or motor drive. In other meaning, it is required 3 motors to drive the 3 degree of freedom of a single leg. The motors used has almost same power rating except the motor of 2nd degree of freedom has higher power rating as it requires higher power and torque to lift up a weight summation of thigh, 3rd degree of freedom motor and lower thigh [1].

2.3 Work Envelope

Work envelope of a robot arm/manipulator/leg is the boundary in space where the tool tip or foot tip can reach [2]. The work envelope of legs can be determined by using graphical analysis provided the angular limit of motors and joints are known.

Forward kinematic method is useful for determining the work envelope as typical leg mechanism resembles serial manipulator [3].

2.4 Gait Consideration

Generally, the term "walking gait" is used in describing legged robot's moving pattern. Gait is literally describing the sequence of the lifting and the placing of the legs and the relative time between these movements [4]. For a quadruped robot, it is able to perform three kinds of gaits, namely crawling gait, trotting gait and a gait that combining crawl and trot [4]. However, the author believes to study and develop all three gaits for quadruped robot might take years to achieve. Therefore, only crawling gait is paid attention in this project due to the time constraint imposed on the final year project. On the other hand, even though crawling gait is relatively slow than other gaits, it still deserves the author's attention because robot's speed is not a focus in this project. On top of that, it complies with the aim of driving the quadruped robot to move as the previous robot by Mr. Tnay Chiat Siang failed to do so. Gait is assumed to be cyclic and only one leg is lifted once per cycle [4]. In other words, legs are lifted up in a manner of one leg per cycle and one leg after another. Such manner of gait is to adjust the body motion in an attempt to maintain its stability. An example of crawling gait sequence is defined in *Appendix 1*.

2.5 Stability Consideration

Stability is an utterly important issue in walking robots. Generally, all the walking robots confront with the problem of what is the suitable distance that each leg transfer in order to maintain its stability [5]. Therefore, how and where does the center of gravity (COG) lie has becoming an important factor in maintain the robot's stability.

There are two types of stability for a walking robot, namely static stability and dynamic stability. Static stability is meaning as the robot's center of gravity (COG) is always within the polygon formed by its supporting leg. The gait performed under such stability condition is called static gait, for instance a walk or crawl [6]. Whereas, the dynamic stability which performing dynamic gait such as a trot or gallop does not necessarily have the center of gravity (COG) lies within the supporting polygon; it involves a much more difficult balancing task [6]. In view of the objective of this project is to have a quadruped robot crawling on flat and horizontal ground. Apparently, it only requires a moderate speed of gait or even slower. Hence, the author will only focus on the static stability of the robot.

The gait of the quadruped robot in this project is technically crawling with moderate moving speed, and thus static stability is emphasized. Static stability margin is defined as the smallest horizontal distance from the center of gravity (COG) to the border of the support polygon [7]. In other words, the margin is a measure of stability where positive stability margin indicates the robot is statically stable while the negative goes the other way round. One important fact is that a legged robot is not statically balanced if fewer than three legs are on the ground. Therefore only one leg is allowed to lift at one time because leaving two legs in the air will not be possible for a quadruped to stabilize. Moreover, to achieve static stability, a legged robot is required to perform gait slowly in order to minimize the body velocity and in turn the inertial force [6].

2.6 Lifting and Landing Timing Consideration

Step 1 in *Appendix 1* is taken as an example in *Figure 2.3* for illustration of timing consideration altogether with static stability margin. *Figure 2.3 (a)* shows that the COG barely laid inside the stability polygon. This happens because red-colored leg lifts up simultaneous with the forward motion of body. The COG lay right on the boundary line of stability polygon at the moment red-colored leg lifts up. Although the COG will be pushed forward by the other legs, the static stability still hard to be secured when red-colored leg lifts up. *Figure 2.3 (b)* shows the COG is already farther from the boundary line of stability polygon when red-colored leg lifts up and travels forward. At the first moment of gait, red-colored leg lay on the ground instead of lifting up. The red-colored leg will only lift up after a short moment of delay in order to avoid tip-over.

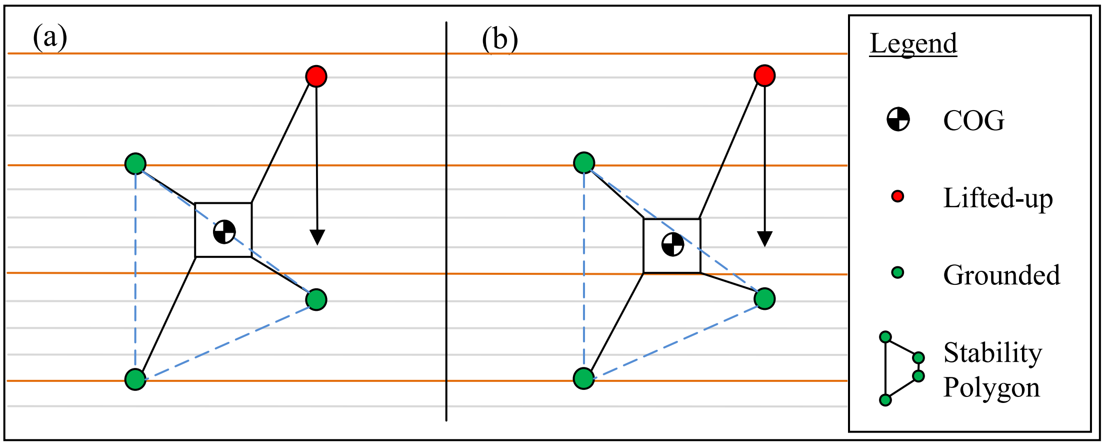


Figure 2.3: Top View of Quadruped Robot with COG inside Stability Polygon. (a) COG barely lay inside stability polygon. (b)COG lay inside stability polygon completely.

2.7 Electronic Hardware

The basic electronic components which are required in such a legged robot project are, for example servomotors, microcontrollers, crystal clock and some other basic electronic components such as resistors and LEDs.

2.7.1 Servomotors

Servomotors are usually used in remote-controlled model such as airplanes, cars, helicopters models and etc. because of its feature of high positioning accuracy yet easy to manipulate [8]. Servomotors are actually DC motors with feedback control system which allows the rotors to be positioned with extremely high accuracy. Due to its internal feedback control system, the position of rotor of a powered-up servomotor will not be altered by external load easily. The feedback control system will automatically adjust the rotor back to the desired position. Therefore, the author find that Proportional-Integral-Derivative (PID) controller is not needed for this project since the servomotors already have its own feedback control system.

Generally, servomotors are able to position at a range of 90 degrees rotation (clockwise 45 degrees rotation or anti-clockwise 45 degrees rotation) depends on the manufacturer's specification. However, the range of position can be extended to 180 degrees by adjusting the control signal. Servomotor has three wires, where two of them are for power of +5V and GND, while the third wire is for position control signal input. The position of the servomotor rotor is controlled by pulse-width modulation (PWM) signal which is a variable width of signal-ON pulse ranged from 1 to 2 milliseconds. A 1 millisecond PWM signal places the rotor at the extreme anti-clockwise position (-45 degrees) while 2 milliseconds PWM signal places the rotor at the extreme clockwise position (+45 degrees). A 1.5 millisecond signal will place the rotor at the neutral position (0 degree). The PWM signal for servomotor is usually configured at 50 to 60 Hz which set the pulses period to be approximate 16 to 18 milliseconds [8]. *Figure 2.4* shows the PWM signal for servomotors.

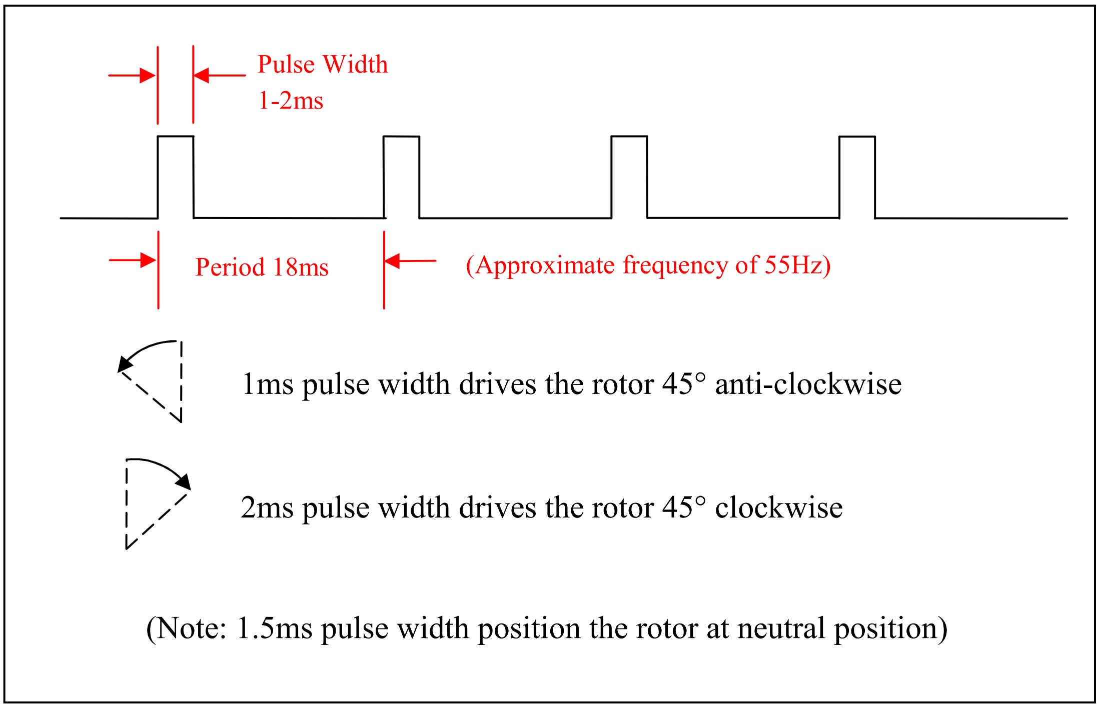

Pulse Width
1-2ms
Period 18ms
(Approximate frequency of 55Hz)
1ms pulse width drives the rotor 45° anti-clockwise
2ms pulse width drives the rotor 45° clockwise
(Note: 1.5ms pulse width position the rotor at neutral position)

Figure 2.4: PWM Signal for Servomotors

2.7.2 Microcontrollers

Microcontrollers are preprogrammed electronic devices that are able to perform particular output based on electronic input [9]. To do so, a microcontroller has to have a CPU (microprocessor) in addition of certain amount of RAM (Random Access Memory), ROM (Read Only Memory), I/O ports and timers on a single chip.

RAM is used by the microcontroller to store the temporary data while the program is running. Higher amount of RAM means higher capability of the chip to process more data during operation. The data stored in RAM will be lost if the power goes off and it is therefore called as volatile memory. ROM is the memory space for storing fixed and permanent data such as application software and this type of memory will not disappear even if the power supply is disconnected [10]. When the microcontroller has CPU and sufficient and necessary memory spaces to operate, it needs something to trigger an expected event. In other meaning, it needs input to prompt the decision making process and output corresponding event. Therefore, I/O pins are essential for microcontroller in order to render something happening. Timer is considered peripheral device of microcontroller in which it exists either internally or externally. The function of timer is to measure time period, determine pulse width and the periodic event [11].

2.7.3 PIC18F Microcontroller

There are quite a number of microcontroller manufacturers in the market, for instance, Microchip, Motorola, Intel and etc. Microcontrollers from Microchip which named as PIC are chosen in this project because of its availability at the electronic store of UTP and most importantly, UTP has the programmer of PIC which allows reprogramming of a PIC. On the other hand, PIC18F series chips are chosen as the letter ‘F’ of PIC18F is an abbreviation of flash which means the PIC18F chips have on-chip program ROM in the form of flash memory. Flash memory is an EEPROM (Electrically Erasable PROM) where it can be reprogrammed easily in seconds and this is the reason why PIC18F series chips are chosen [10].

2.7.4 Oscillators

Oscillator is a processor clock that sequences the instructions within the CPU of microcontroller and provides time base for peripheral application. The range of frequency starts from as low as few kilohertz to as high as hundred megahertz depends on the specification of the microcontroller. As for PIC18F458, it is able to operate at a maximum frequency of 40 MHz. Generally, PIC microcontrollers can be operated in four different oscillator modes, i.e. low-power crystal (LP), crystal/resonator (XT), high-speed crystal/resonator (HS) and resistor/capacitor (RC). LP, XT and HS modes are all using crystal or ceramic oscillator with certain range of frequency. Crystal and ceramic oscillator are both having satisfactory timing accuracy. RC oscillator, being the remaining mode of oscillator, is only consisting of resistor and capacitor. This turns out to be extremely low cost. However, such oscillator has low timing accuracy [8].

2.8 Pulse-Width Modulation

Pulse-width modulation (PWM) is a method to provide digital to analog signal conversion. PWM is a series of pulses in which the duty cycle (logic high) of a square wave output is varying to provide a varying average DC output voltage [11]. In other words, PWM is used by microcontroller to output average DC voltage to analog device as microcontroller itself is only capable of operating in digital signal (logic high or low). An example illustrates PWM is shown in *Figure 2.5*.

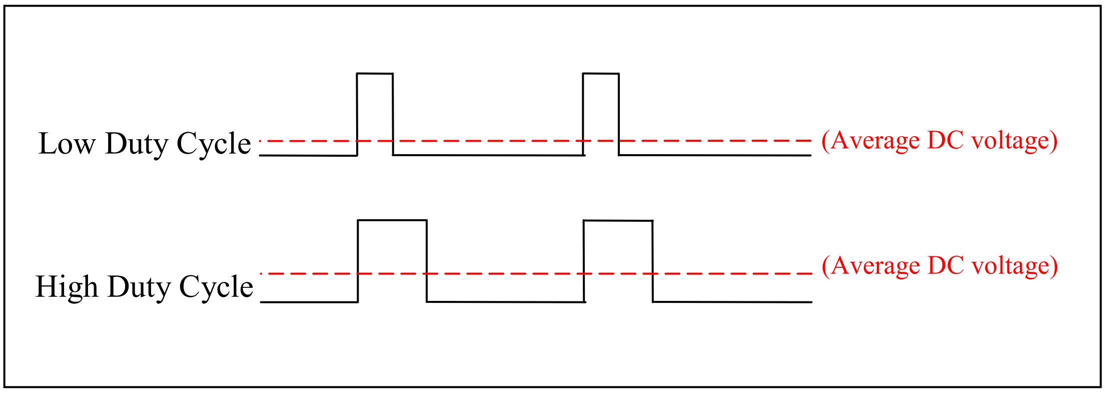


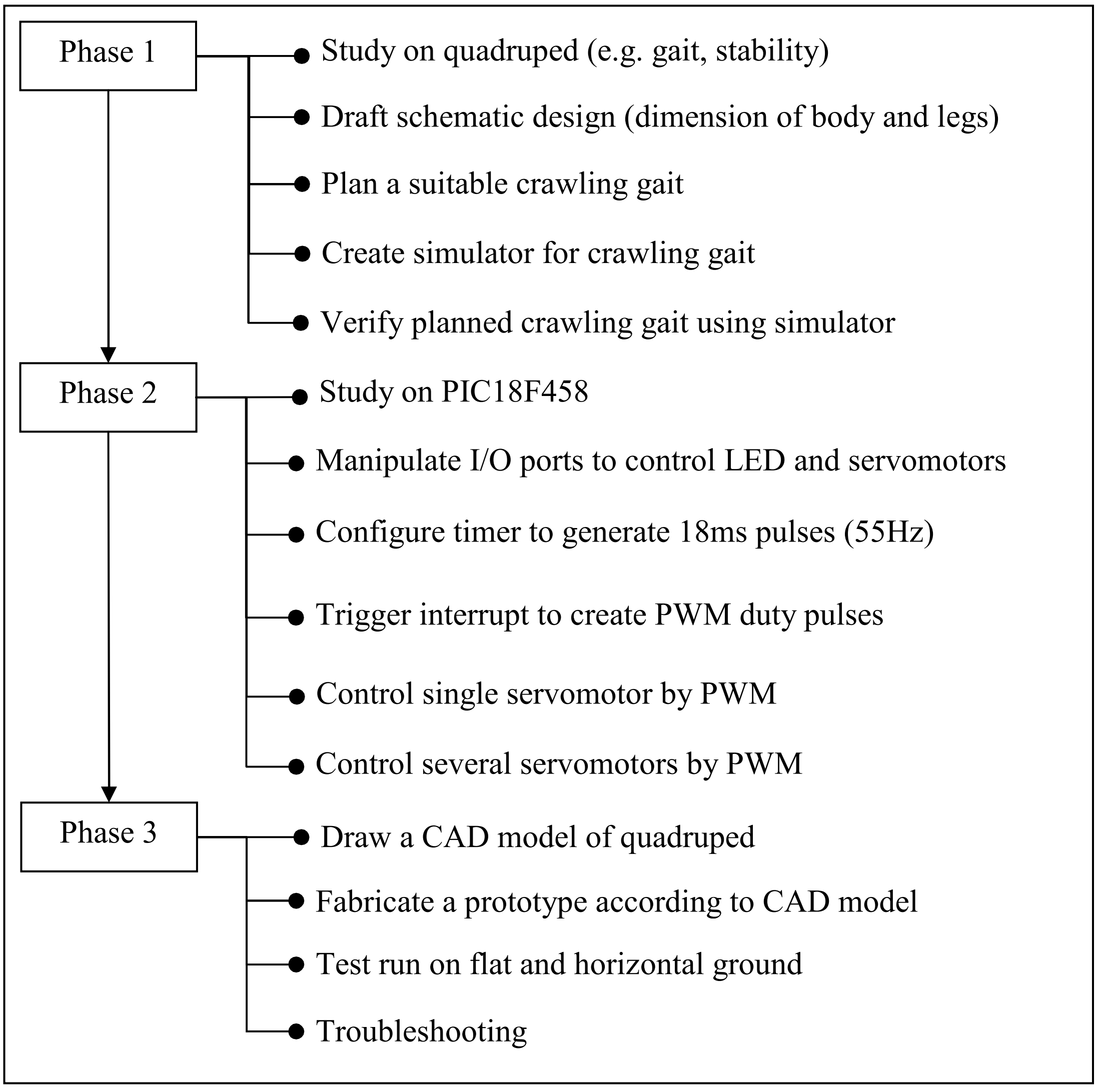
Figure 2.5: Pulse Width Modulation as Average DC Voltage

CHAPTER 3

METHODOLOGY

3.1 Project Phase

Phase 1 of the project was completed in last semester while phase 2 is currently underway in this semester. Phase 3 is supposed to be completed by the end of FYP 2.



Phase 1
Study on quadruped (e.g. gait, stability)
Draft schematic design (dimension of body and legs)
Plan a suitable crawling gait
Create simulator for crawling gait
Verify planned crawling gait using simulator
Phase 2
Study on PIC18F458
Manipulate I/O ports to control LED and servomotors
Configure timer to generate 18ms pulses (55Hz)
Trigger interrupt to create PWM duty pulses
Control single servomotor by PWM
Control several servomotors by PWM
Phase 3
Draw a CAD model of quadruped
Fabricate a prototype according to CAD model
Test run on flat and horizontal ground
Troubleshooting

Figure 3.1: Project Phase Chart

3.2 Phase 1

3.2.1 Flow Chart

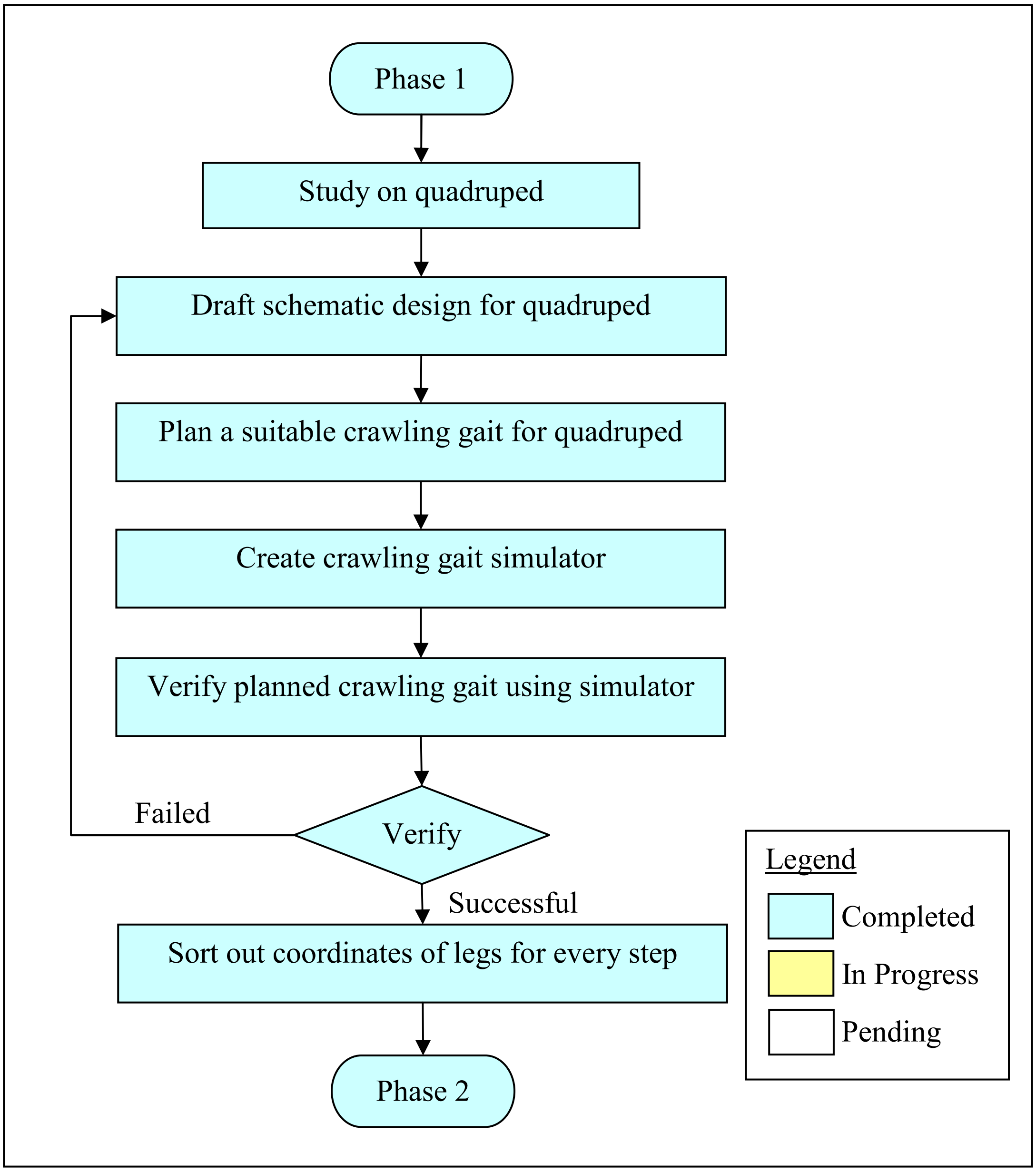

Phase 1
Study on quadruped
Draft schematic design for quadruped
Plan a suitable crawling gait for quadruped
Create crawling gait simulator
Verify planned crawling gait using simulator
Failed
Verify
Successful
Sort out coordinates of legs for every step
Phase 2
Legend
Completed
In Progress
Pending

Figure 3.2: Phase 1 Flow Chart

3.2.2 Schematic Design

A schematic design of quadruped robot is drafted at first in order to have a set of preliminary dimensions of the body and legs. CAD model is not started yet at this point as the specifications of quadruped are not finalized. On top of that, schematic design is preferable instead of CAD model at this stage because it is not a wise act to fix the dimensions without taking simulation result into consideration. CAD drawing will only be started after all the completion of mechanical simulation and wiring, circuit-board design. Moreover, schematic allows changes and modifications if problems incur the mechanical simulation. The schematic design of quadruped robot in top view and right-hand side view are defined in *Appendix 2* and *Appendix 3* respectively.

3.2.3 Coordinate and Joint Angle Notation

Coordinate of each joints are labeled with specified notations for the sake of reading and tracking. Such notations are named as global coordinate notation as all joints are referring to common reference axes, namely x-axis, y-axis and z-axis. Each joint consists of x-axis, y-axis and z-axis coordinates with COG of quadruped defined as the origin of all axes, i.e. (0. 0. 0). The global coordinate notation of quadruped in top view is defined in *Appendix 4*.

Joint angle always refer to a reference axis. Joint angles of quadruped in top view and right-hand view are defined as per *Appendix 5* and *Appendix 6* respectively. The reference axis is where the arrow starts from. The heading direction of each arrow indicates the positive value of angle while the negative value goes inversely.

3.2.4 Calculation of Joint Angle and Position

Joint angle can be obtained by using *Trigonometry* and *Law of Cosines* provided the positions of joint 2 and foot tip are known (position is named as coordinate from here on since the position of joints or foot tips are represented by global coordinate notation). Same method is used to calculating joint 3 coordinate if the coordinates of joint 2 and foot tip are known. An example of joint angle and coordinate calculation is shown in *Appendix 7*.

3.2.5 Crawling Gait Planning

A crawling gait of 8 steps is explained previously in *Section 2.4* and the author does mention that the number of steps can be any number of common factors of four with condition no less than eight. Therefore, the author has come up with a 16 steps crawling gait which has higher resolution (meaning more detailed movements are shown). Static stability margin is the main issue during the planning of crawling gait in which COG has to be constraint within the boundary of stability polygon. Other than that, the timing of leg lifting and landing mentioned in *Section 2.6* is taken into account for the gait planning. The crawling gait of 16 steps with adjusted-timing is defined in *Appendix 8*.

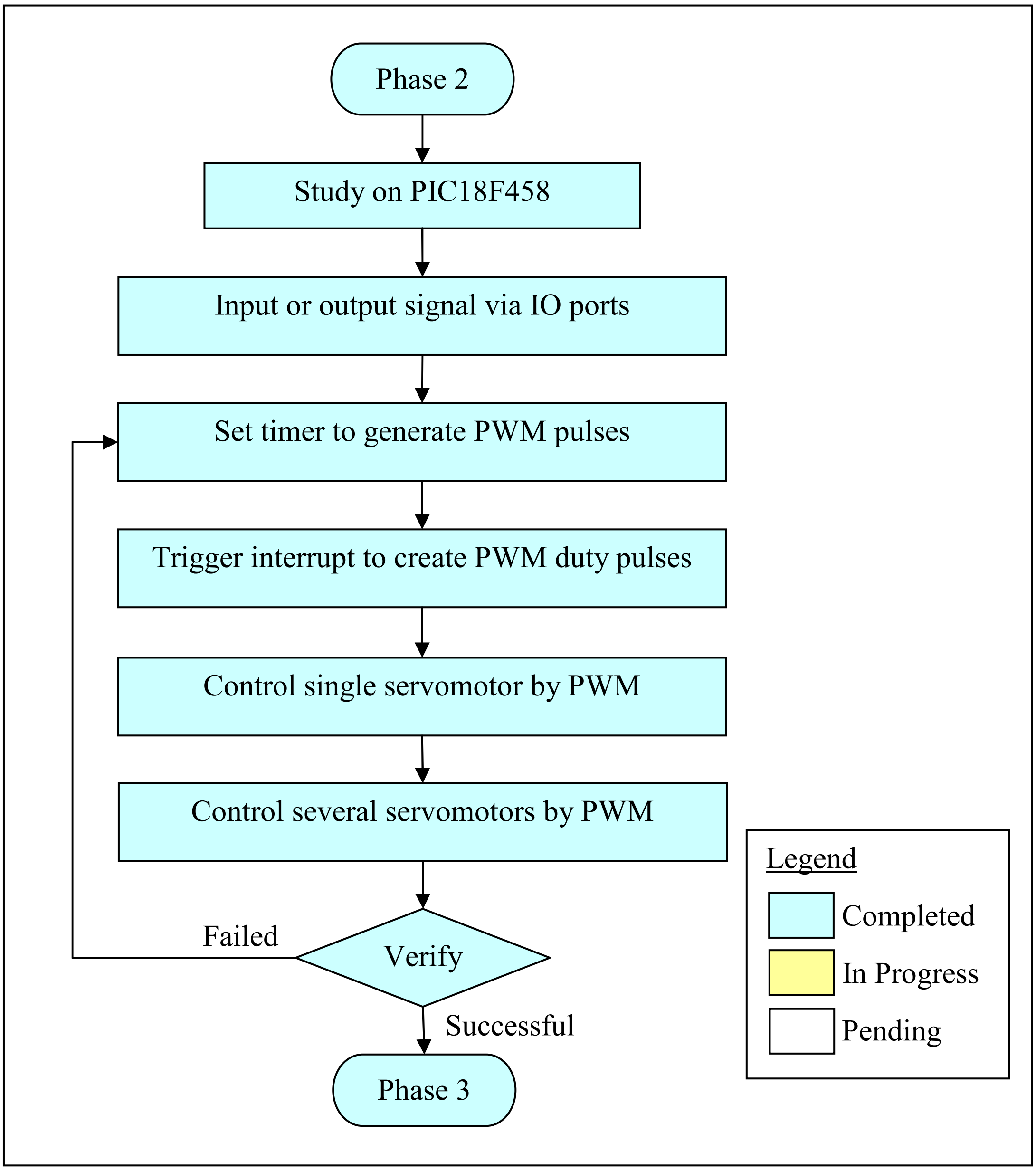
3.2.6 Creating Preliminary Simulator

The simulator is addressed as preliminary simulator is because the parameters used in this simulator is somewhat ‘immature’ which means the dimensions of quadruped are still in schematic design stage and are not finalized yet. The preliminary simulator will simulate the crawling gait cycle of 16 steps with adjusted timing of leg lifting and landing. The result of preliminary simulator will be discussed in Chapter 4.

3.3 Phase 2

The focus of phase 2 is on the electronic part involves programming the PIC microcontroller and control the servomotors according to the gait planned in phase 1.

3.3.1 Flow Chart



Phase 2
Study on PIC18F458
Input or output signal via IO ports
Set timer to generate PWM pulses
Trigger interrupt to create PWM duty pulses
Control single servomotor by PWM
Control several servomotors by PWM
Failed
Verify
Successful
Phase 3
Legend
Completed
In Progress
Pending

Figure 3.3: Phase 2 Flow Chart

3.3.2 Hitec HS-322HD Servomotor

The servomotors used in this project are HS-322HD servomotors from Hitec. This series of servomotors are designed for hobbyist's application such as model aircraft, steering and throttle control for model car and etc. This motor operates under the same concept mentioned in *Section 2.7.1* in which the rotor position +/- 45 degrees by manipulating the pulse width from range 1 to 2 milliseconds. However, according to the official specification from website this motor is able to extend its working range by extending to +/- 90 degrees when given a pulse ranged ranging from 0.6 to 2.4 milliseconds. The operating voltage for HS-322HD is approximate 4.8 to 6.0 V with 4.8 V generate output torque of 3 kg.cm while 6.0 V generate output torque as high as 3.7 kg.cm. An example of HS-322HD servomotor is shown in *Figure 3.4*.

Figure 3.4: Hitec HS-322HD Servomotor

3.3.3 PIC18F458

Based on the available data sheet (released by Microchip Technology, Inc.) and reference book, i.e. the book titled "*Embedded C Programming and the Microchip PIC*" from Thomson Learning, Inc., PIC18F458 are used in this project for servomotors control. According to its data sheet, PIC18F458 is available in 28-pins, 40-pins and 44-pins. It has 32k bytes of internal program memory, 1536 bytes of RAM memory and 256 bytes of EEPROM memory. Other than that, it also has four internal timers with selectable pre-scaler. It also provides two

Capture/Compare/PWM modules to capture and compare the timing when external event occur or send out PWM signal. The pin diagram of PIC18F458 is shown in *Figure 3.5*.

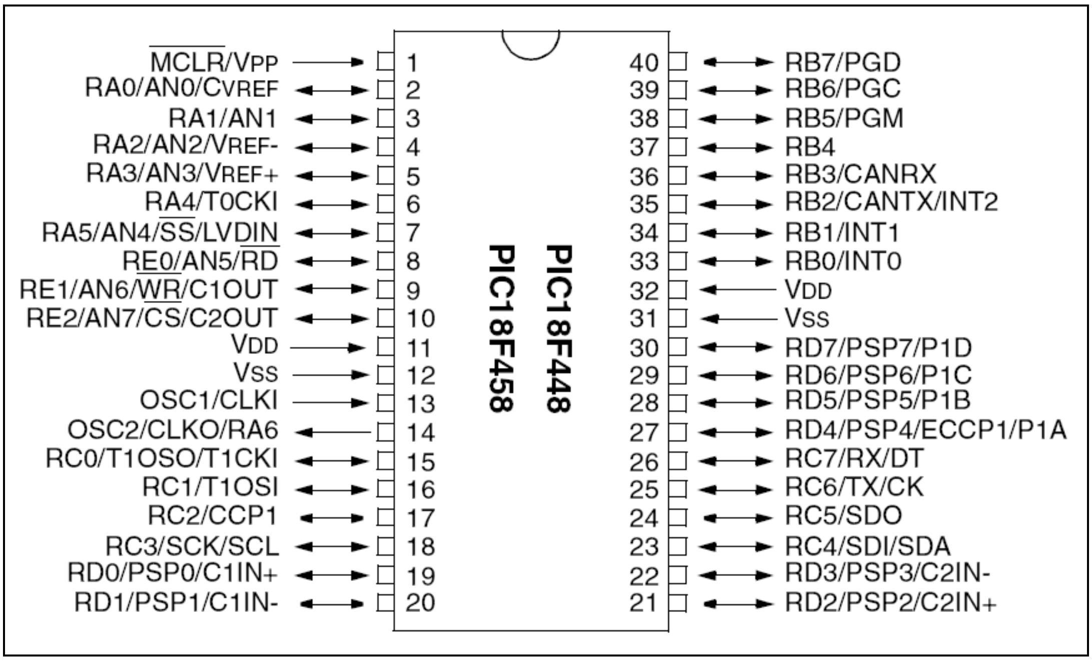


Figure 3.5: Pin Diagram of PIC18F458

3.3.4 Software and Hardware

As for the software, PICC C Compiler by CCS, Inc. is used to compile program in C language and generate hex-code (hexadecimal numbers) file for re-programming (burning) the PIC microcontrollers. Other than that, MPLAB IDE from Microchip Technology, Inc. which is also compatible with PICC C Compiler is used to simulate and troubleshoot the program before the hex code file is burned into the PIC microcontroller.

To burn the program into the PIC microcontroller, WARP13 programmer board is used. WARP13 programmer board is able to erase and re-write the EEPROM memory of PIC14F/16F/18F series microcontroller. What WARP13 programmer needs is only a hex-code file which generated by PICC C compiler to program a PIC Flash series microcontroller. Assembly language needs not be bothered because MPLAB IDE is able to convert the C language program into assembly language program based on the hex-code file generated by PICC C Compiler. Note that microcontroller recognizes hex-code only whereas C language and assembly language are recognized by human and this is the reason why a compiler is needed (to convert either C language program or assembly language program to hex-code).

3.3.5 Constructing C Language Program

The PICC C Compiler is very much the same with other C language compiler, with additional microcontroller-specified function library. To create PWM output signal from PIC18F458, at least one internal timer and relevant interrupt function is needed. The concept of the program is enabling timer to run once the microcontroller is powered up and let the timer counts 18 ms (there are delay functions within the library of PICC C Compiler). When the timer's count reach 18ms, an interrupt function is prompted and output logic high at pre-set pins/ports for certain period (in this case, 1 to 2 milliseconds duty cycle is used for servomotors control). When the selected pin/ports output logic low, the timer will be reset and run for another 18 millisecond which then prompt interrupt function again and so on. By varying the duty cycle time (the pulse width of logic high), the position of the rotor can be varied. The concept of creating PWM in PIC18F458 is illustrated in flow chart as shown in *Figure 3.6*.

When the microcontroller executes the main program, it is possible to interrupt the main program and executes something else by enabling interrupt function. The microcontroller will resume or return to the breakpoint where it had stopped after the tasks in the interrupt function are completed. Such function therefore makes the application to be more real-time operating because interrupt function allows the microcontroller to respond to changes of hardware environment. By a look at the data sheet of PIC18F458, it has 21 interrupt sources relevant to timers, external input, ADC and etc.

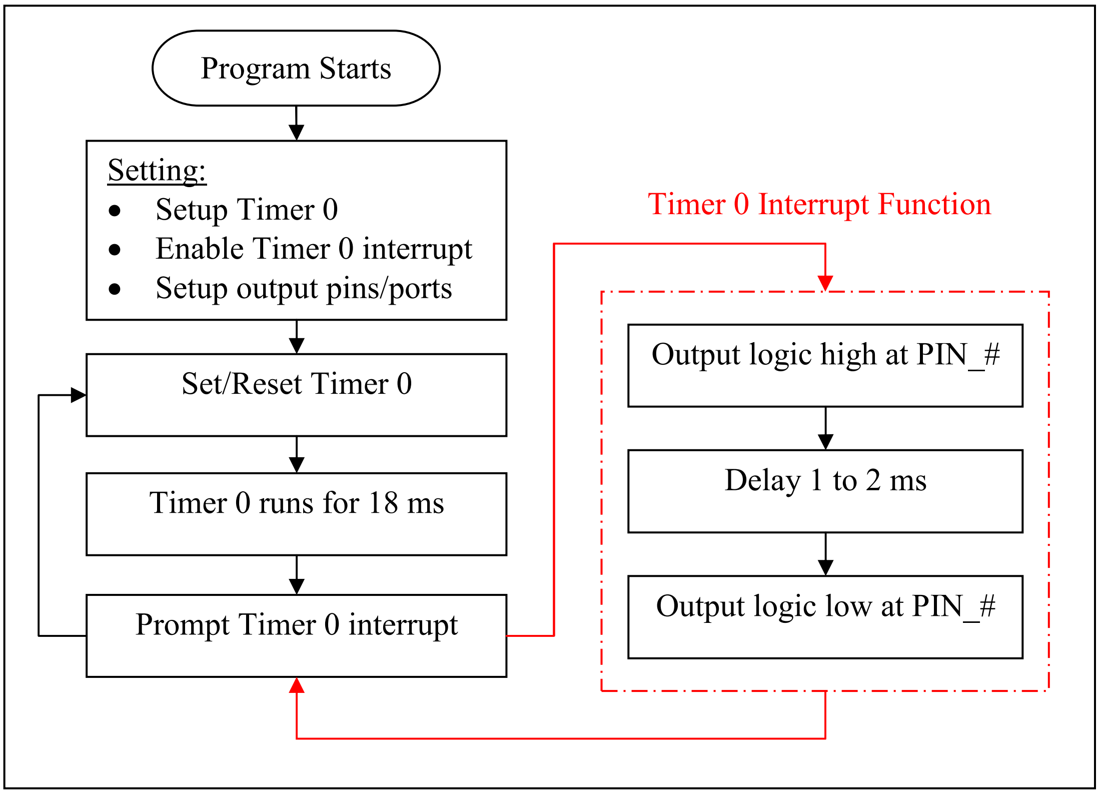


Figure 3.6: Flow Chart of PWM Output by Using Timer 0 and Interrupt

3.3.6 Single Servomotor

To learn about the PWM technique using PIC18F458, an example program is constructed by the author and it has been tried out on with a servomotor. The single servomotor program is shown in *Appendix 9*. The lesson of this program is to find out how exactly a PIC18F458 send out PWM signal to servomotor. There are three possible techniques of creating PWM in PIC18F458, namely using internal CCP module, for loop and timer interrupt. Fortunately, one of the possible techniques, and the only one which is proved to be working well with the Hitec HS-322HD servomotor is using timer interrupt.

The problem of internal CCP module is because of the pulse period generated is too small for the required frequency of servomotors, i.e. 50 to 60 Hz (approximate period time of 16 to 20 milliseconds). For loop technique is not a viable option either because of the inaccuracy of the timing, which meaning the pulse width and pulse period are difficult to control. Explanation in detail why these two techniques are not chosen will not be demonstrated as the author’s interest is solely laid on the viable solution.

Timer interrupt, being the only viable technique is used to send out PWM signal. Refer to *Appendix 9*, RTCC timer (same with timer 0) starts counting up until 65535 ($FF FF in hexadecimal number) where RTCC interrupt function will be triggered. Output pin is set logic high for 1 ms (*syntax: delay_ms(1);*) at the beginning of RTCC interrupt function and the logic high status will be extended for more time depends on the duty value. For instance, duty value of 15 means the logic high status will be extended for 500 μs more (*syntax: delay_us(500);*). Output pin will be set logic low before the ending of RTCC interrupt function. Note that RTCC timer is reset at 47535 ($B9 AF in hexadecimal number; *syntax: set_rtcc(47535);*) which means the RTCC timer will start counting from 47535 to 65535 where RTCC interrupt function will be triggered again and so on.

The reason of resetting RTCC timer at 47535 is because counting from 47535 to 65535 takes 18000 instruction cycles to complete, and this 18000 instruction cycles are just nice to create 18 milliseconds of pulse period. Note that the clock frequency is 4 MHz which means each instruction cycle takes 1 μs (clock frequency of 4 MHz shall be divided by 4 according to data sheet).

Simulation of the single servomotor program is done in MPLAB IDE and the logic analyzer tracks a series of pulses are generated successfully as shown in *Figure 3.7*. The program is justified at this moment from the software-wise. However, it needs to be justified with the hardware as well. Fortunately, the program is proven to be working as expected when a programmed PIC18F458 is connected to the wiring diagram as shown in *Appendix 10*.

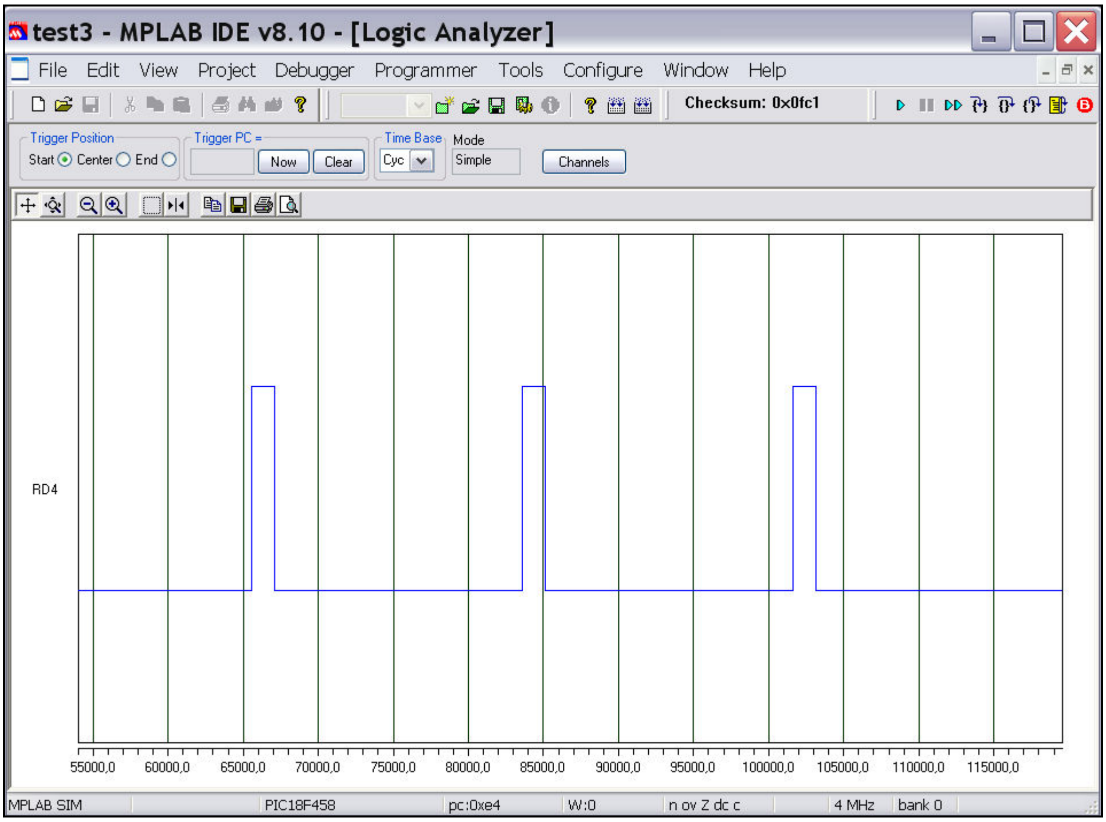


Figure 3.7: Simulation Result of Single Servomotor

3.3.7 Several Servomotors

After the completion of single servomotor program, controlling several servomotors (three servomotors in this case) has been tried out by the author. The concept of this program is very much the same with the single servomotor program as mentioned in *Section 3.3.6*, the different part of this program is the implementation of timer 0, timer 1 and timer 3 altogether. The simulation result is shown in *Figure 3.8*. The program is shown in *Appendix 11* and the wiring diagram is shown in *Appendix 12*.

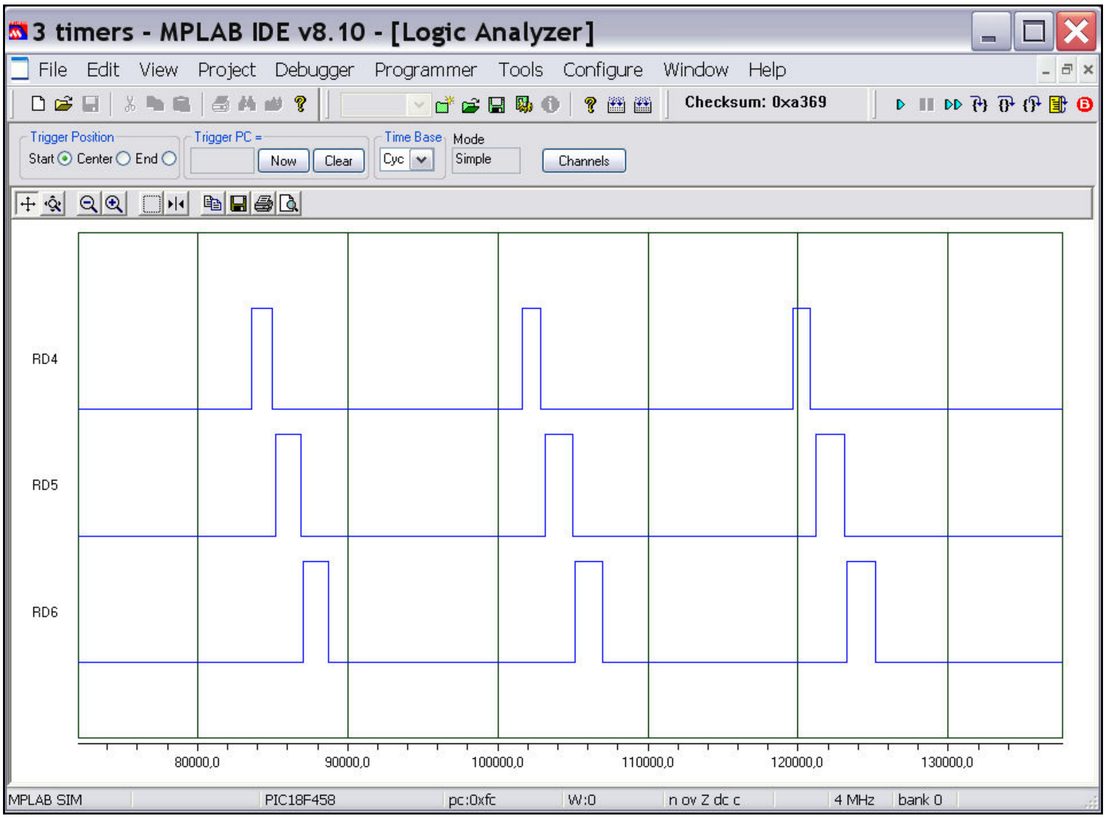


Figure 3.8: Simulation Result of Three Servomotors

3.4 Phase 3

Phase 3 is the last phase of the entire project which basically involves fabrication, testing and troubleshooting. The reason of placing CAD and prototyping in the last phase is that there are too many uncertainties and modifications in phase 1 and phase 2, such as robot body size might change to fit in the circuit board and motors, leg dimension might change to ensure COG lay within stability polygon and etc. All these uncertainties will only be clarified after the completion of phase 1 and phase 2.

After prototype is built, motors will be mounted onto the prototype and the simulator will be readjusted by using the finalized parameters (leg dimensions). At this point, the quadruped prototype shall be acting as expected as the result from simulation. Any deviation from expectation shall be investigated and analyzed.

3.4.1 Flow Chart

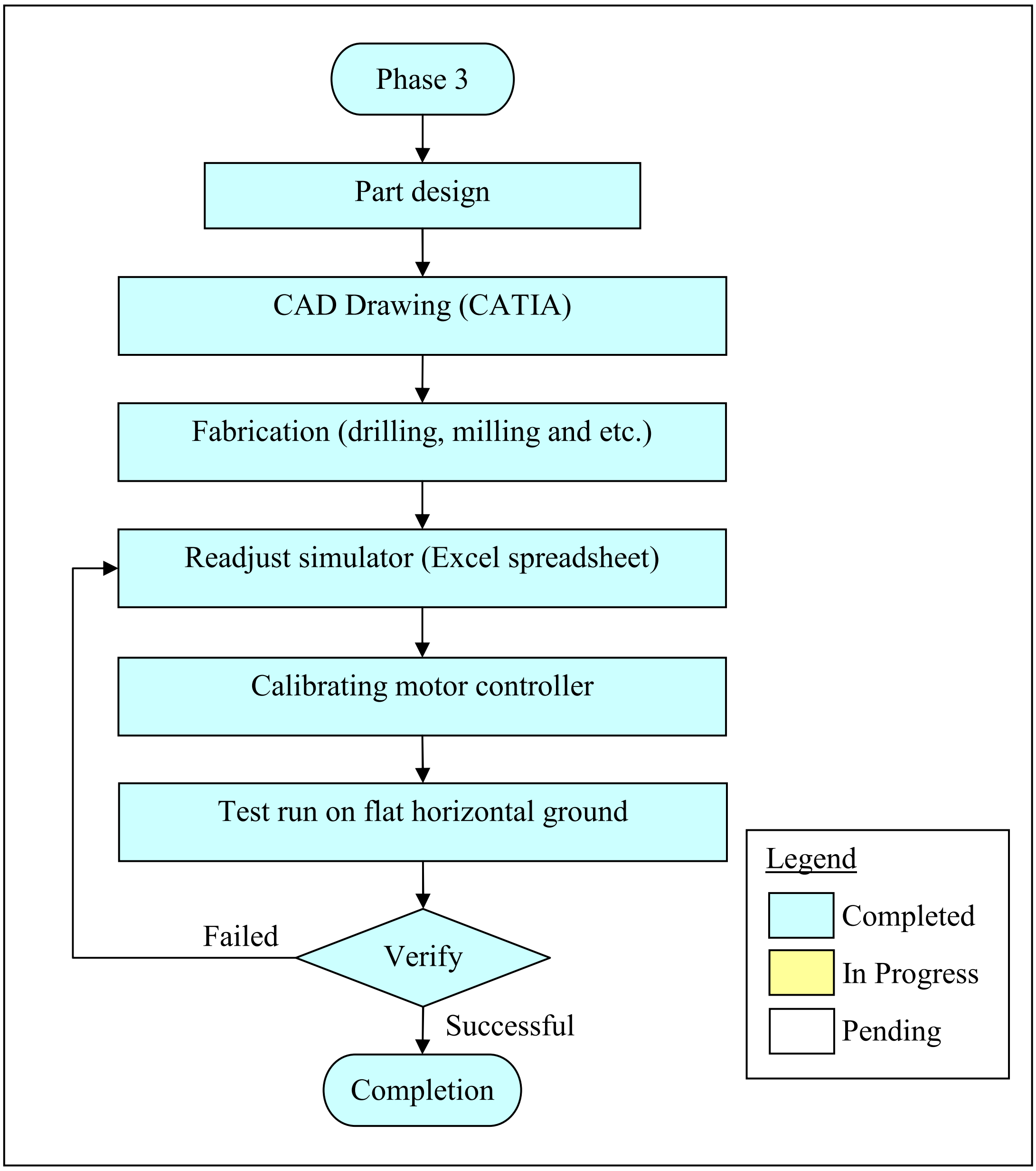

Phase 3
Part design
CAD Drawing (CATIA)
Fabrication (drilling, milling and etc.)
Readjust simulator (Excel spreadsheet)
Calibrating motor controller
Test run on flat horizontal ground
Failed
Verify
Successful
Completion
Legend
Completed
In Progress
Pending

Figure 3.9: Phase 3 Flow Chart

3.4.2 CAD Drawing

Leg dimensions are clarified prior to the commencing of CAD drawing. Consideration such as availability of material and machine, method of fabrication and complexity of the part design are taken into account so that the design of prototype will not lead to any difficulty of material preparation and parts fabrication.

Motor brackets and linkages are standardized for all twelve motors in order to reduce the fabrication time and complexity. Other than that, such approach is able to reduce possibility of dimension error as less part design will be less error. The technical drawings of quadruped are in *Appendix 13*.

3.4.3 Fabrication

Fabrication is currently underway and most of the parts are nearly finished fabricating. The material used for this prototype is aluminium plate because of its characteristic of light weight and considerable strength. The prototype basically consists of body frame, motor bracket, linkages and all four legs are sharing the same design which means each leg has same number of motors, motor brackets and linkages.

The methods of fabricating the aluminium plates into particular parts involve shearing, bending, milling and drilling. Shearing is used to cut a large piece of aluminium plate into small pieces or raw plate of each part. Bending is a process to bend a flat plate by certain angles. Milling and drilling are the processes of removing material by using special cutting tools such as drill bit and carbide. The process details of each part are shown in *Table 3.1*.

Table 3.1: Part Fabrication Details

Parts	**Description**	**Processes**
Motor bracket (Qua-J1)	Bracket of holding motor and acts as joint	Shearing
		Drilling
		Milling
		Bending
Linkage type-I (Qua-L2)	Link joint 2 to joint 3 and acts as thigh	Shearing
		Drilling
		Milling
		Bending
Linkage type-II (Qua-L3)	Extend from joint 3 to foot tip and acts as lower thigh	Shearing
		Drilling
		Milling
Body frame (Qua-F1)	Body of quadruped and this is where the COG lies.	Shearing
		Drilling
		Milling

3.4.4 Readjusting Simulator

The spreadsheet simulator has to be readjusted by using the real parameters of the prototype because the dimensions of the prototype are different from the preliminary simulator which is developed based on the schematic design. The purpose of the readjusted simulation is to provide a close-to-reality gait pattern for quadruped in which the prototype will perform crawling gait according to simulation. Therefore, the dimensions of the prototype have to be taken into account when readjusting the simulator.

On the other hand, timing adjustment has to be made as calibrating and test run goes because the timing consideration in *Section 2.6* is yet to be proven with the prototype. The lifting and landing timing adjustment mentioned in *Section 2.6* (a short moment of delay or advance) might have to be scrapped and implement the conventional crawling gait as shown in *Appendix 1* if the stability of prototype is jeopardized during performing crawling gait.

3.4.5 Calibrating Motor Controller

Based on the result of readjusted simulation mentioned in *Section 3.4.4*, all joint angles and coordinates shall be implemented in the motor controller programs. The motor controller programs consist of PWM duty cycles of all servomotors which are corresponding to the joint angles throughout the entire cycle of crawling gait. The result of the calibrated motor controller will be discussed in Chapter 4.

3.4.6 Test Run of Prototype

Prototype equipped with motor controller will first be tested by elevating the body frame which means all four legs are placed above the ground. Such test is to verify if the crawling gait of prototype is parallel to the readjusted simulation under condition of no load (by its body weight and reaction force upon the foot tip). If the crawling gait of prototype performs as what expected from the readjusted simulation, the test shall be carried out on the ground. In case of any failure, i.e. tip over, unable to lift up its own body and etc., investigation and troubleshooting shall be started from the readjustment of simulator and calibrating of motor controller.

3.5 Gantt Chart

3.5.1 Gantt Chart of FYP 1

No	Detail/ Week	1	2	3	4	5	6	7	8	9	10		11	12	13	14
1	Selection of Project Topic	■	■									Mid-Term Break				
2	Preliminary Research Work • Research on quadruped		■	■												
3	Submission of Preliminary Report				●											
4	Phase 1: • Study and learning on implementing PIC MCU				■	■	■	■	■							
5	Submission of Progress Report								●							
6	Seminar								●							
7	Phase 1: • Schematic design • Gait simulation									■	■		■	■	■	■
8	Submission of Interim Report															●
9	Oral Presentation															●

3.5.2 Gantt Chart of Semester Break

No	Detail/ Week	1	2	3	4	5	6	7
1	Phase 2 (1st Dec 08 – 20th Dec 08): • Study on PIC microcontroller	■	■	■	BREAK			
2	Phase 2 (28th Dec 08 – 17th Jan 09): • Learning to manipulate PIC microcontroller					■	■	■

3.5.3 Gantt Chart of FYP 2

No	Detail/ Week	1	2	3	4	5	6	7	8	9		10	11	12	13	14
1	Phase 2: • PWM control • PIC communication	■	■	■	■						Mid-Term Break					
2	Submission of Progress Report 1				●											
3	Phase 3: • CAD design • Fabricating prototype					■	■	■								
4	Submission of Progress Report 2								●							
5	Seminar									●						
6	Phase 3: • Testing								■	■		■				
7	Submission of Poster											●				
8	Dissertation (Soft bound)														●	
9	Oral Presentation															●
10	Submission of Project Dissertation (Hard bound)															●

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Preliminary Simulation (Phase 1)

Microsoft EXCEL spreadsheet is used to create a crawling gait simulator. The purpose of the preliminary simulator is to provide a basic concept of quadruped crawling gait with adjusted timing of leg lifting and landing which is elaborated in *Section 2.6*. Note that the result of the preliminary simulation is not yet finalized and some modifications might have to be made in latter stage.

4.1.1 Crawling Gait Analysis

Joint coordinates and joint angles of each step are calculated based on *Trigonometry* and *Law of Cosines* as shown in *Appendix 7*. The crawling gait result of preliminary simulation is defined in *Appendix 14*. Note that apart from the joint coordinates of each step, static stability of quadruped is demonstrated in the simulation as well.

4.1.2 Static Stability Analysis

Having the COG lay within the boundary line of stability polygon indicates the quadruped is balanced / stabilized statically. Planning crawling gait without considering static stability will not drive the quadruped far even if it is able to move. Quadruped with the COG out of the stability polygon will bound to tip-over. Therefore, static stability is taken into account when the simulation of crawling gait is created. The static stability result of preliminary simulation is defined in *Appendix 14*.

4.1.3 Timing Analysis

Timing consideration of quadruped has been discussed in *Section 2.6*. The leg lifting and landing action of some steps are either delayed or brought ahead in the simulation. The leg lifting action of step 1 and step 9 are delayed in order to push the COG farther from the boundary limit of the stability polygon. Whereas, step 7 and step 11 has same approach of ahead leg landing action to avoid the COG closing to the boundary limit of the stability polygon. All the respective approaches are defined in *Appendix 14*.

4.2 Prototype (Phase 3)

A prototype is constructed according to the CAD model. The fabrication processes of various parts are defined in *Table 3.1* and the technical drawings of the CAD model are defined in *Appendix 13*. The motor-mounted prototype without circuit board is weighted 1547 gram as measured by digital weight indicator. The image of quadruped prototype is shown below.

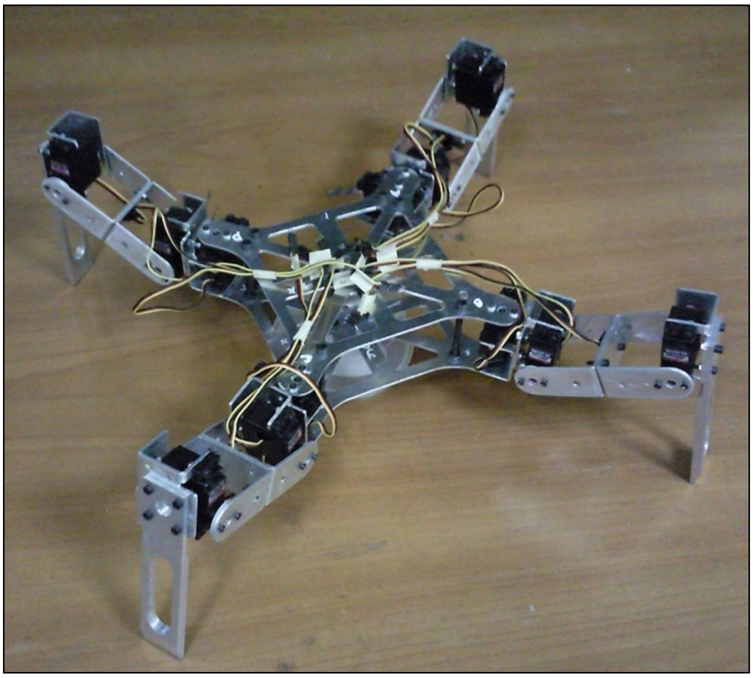


Figure 4.1: Quadruped Prototype

4.3 Readjusted Simulation (Phase 3)

The preliminary simulation is readjusted in phase 3 by referring to the real dimensions of the prototype. The result of the readjusted simulation is a basic structure for the motor controller and it provides the joints angles, or motor rotational angles that are required in the motor controller. The result of readjusted simulation is defined in *Appendix 15*.

4.3.1 Crawling Gait Analysis

Joint coordinates and joint angles of each step are calculated according to *Trigonometry* and *Law of Cosines* as shown in *Appendix 7*. The crawling gait result of readjusted simulation is defined in *Appendix 15*. Note that apart from the joint coordinates of each step, static stability of quadruped is demonstrated in the simulation as well.

4.3.2 Static Stability Analysis

Stability means the world to quadruped and it can be achieved only if the COG lies within the boundary line of stability polygon. The static stability result of readjusted simulation is defined in *Appendix 15*.

4.3.3 Timing Analysis

Based on the result of readjusted simulation, the timing consideration of quadruped which has been discussed in *Section 2.6* is not necessary for the prototype because the COG of prototype is still lying within the static stability polygon even though the adjusted timing of leg lifting and landing mentioned in *Section 2.6* is not implemented. Therefore, the adjusted timing of leg lifting and landing in preliminary simulation is scrapped and the conventional crawling gait as shown in *Appendix 1* is implemented instead. The timing of leg lifting and landing result is demonstrated in *Appendix 15*.

4.4 Motor Controller (Phase 3)

To drive the prototype performs crawling gait, motor controller consists of both software (i.e. program) and hardware (i.e. wiring) has to be designed to control all twelve servomotors. The motor rotational angles are referring to the joint angles of respective step in the readjusted simulator which was elaborated in *Section 4.3*.

4.4.1 Program

Program is the software, or 'brain' of the motor controller that determines what action will be performed by the quadruped. The program is written in C language by using PICC C Compiler. There are two set of programs, namely program of leg A and leg B, and program of leg C and leg D. This is meaning that there are two microcontrollers where each microcontroller controls six servomotors. The reason of using two microcontrollers is because if only one microcontroller is used to control all twelve servomotors, the timing of steps appears to be fairly inaccurate. The explanation of this is probably because of the timer interrupt functions are being triggered too often (since there are twelve servomotors). Therefore, the author decided to use two microcontrollers (two sets of programs) so that the timer interrupt functions will only be triggered at moderate rate. The motor controller programs are defined in *Appendix 16*.

Refer to *Appendix 16*; there is a look-up table in each program. The purpose of look-up table is providing a database of motor rotational angles of each leg at respective steps. The four-digit number is equivalent to thousand microseconds. For instance, a number of 1500 represents 1500 microseconds (1.5 milliseconds) where it will set the rotor of servomotor at neutral position (0 degree). Number larger than 1500 (1.5 milliseconds above) means clockwise rotation (positive degrees) whereas number smaller than 1500 means anti-clockwise rotation (negative degrees).

Note that timer RTCC and TIMER1 are both driven by internal clock of 4 MHz, while TIMER3 is driven by external clock of 4 MHZ with prescaler of 8.

Timer RTCC and TIMER1 are used to generate the periodic pulses of PWM signal at rate of 51 Hz. Timer TIMER3 is used to provide time base of 8 seconds period for the crawling gait (0.5 milliseconds per step).

4.4.2 Wiring

Wiring is the hardware, or 'body' of the motor controller that receives power and delivers power or signal to assorted components. The wiring diagram of motor controller is defined in *Appendix 17*.

There are two microcontrollers (PIC18F458); one for controlling leg A and B (6 servomotors) while another one is for controlling leg C and leg D (6 servomotors).

Two 4 MHz crystals (oscillator) are used in the motor controllers; one is used to provide time base for timer RTCC and TIMER1 (both internal clocks) of both microcontrollers while another one is for timer TIMER3 (external clock).

Besides, two power regulators (78L05) are used in order to provide steady power supply of 5 volt to each microcontroller. Power regulating always lead to heat accumulation at the component, hence, heat sinks are attached to the power regulators in order to dissipate heat and prevent component from failure.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

The project objective is to design a control system for a quadruped robot (4-legged robot) that built by an UTP graduate; Mr. Tnay Chiat Siang (Matric No. 6114). However, the mechanical design by Mr. Tnay has been revised and modified by the author because the prototype built by Mr. Tnay tends to have some mechanical defects, for instance jamming at knee joint, bending at thigh (link 2). Such problems are due to the parallel link design in Mr. Tnay's prototype. To avoid the occurrence of same problem, the author decided to adopt serial-manipulator-like leg mechanism for his project because such design has no joint limitation as parallel link does but at the same time has less link support (lower joint strength).

At the end of project, a new prototype of quadruped is successfully built along with motor controllers which are developed according to the readjusted spreadsheet simulator. The motor controllers are able to drive all twelve servomotors rotating to the respective angles of each step. By experiment, the quadruped is able to perform forward crawling gait on flat and horizontal ground and thus the objective of project is met.

5.2 Recommendation

Since the quadruped prototype in this project is not able to crawl or even stand on its own, troubleshooting has been carried out by the author and there are some recommendations for future reference.

Servomotors of higher torque shall be used in order to withstand the weight of the prototype. When the motor controllers of quadruped prototype are first powered up, the quadruped prototype is supposed to be standing with its four legs and each servomotor shall provide sufficient torque to withstand the body weight. However, the result of test run failed meeting up the expectation and therefore the author assumes lack of motor torque could be the reason.

Other than that, weight reduction could be critical as well. The lighter the prototype is, the lesser the power required or consumed. This can be achieved by drilling some holes on the prototype as long as the holes are not placed at the area of high stress concentration. Besides, the material of bolts and nuts do contribute to the weight of prototype to some extent. Therefore, switching to aluminium bolts and nuts instead of iron bolts and nuts could be a good approach to reduce weight.

If the quadruped prototype were to crawl on flat and horizontal ground by its own, it is definitely a good move to try turning gait or even gait on inclined surface or rugged surface.

REFERENCES

[1] T. W. Tee, K. H. Low & H. Y. Ng. 2002, "*Mechatronics Design and Gait Implementation of a Quadruped Legged Robot",* Nanyang Technological University, Singapore

[2] Ashfal, C. Ray. 1992, *Robots and Manufacturing Automation*, John Wiley & Sons, New York

[3] W. S. Mark, M. Vidyasagar. 1989, *Robot Dynamics and Control*, John Wiley & Sons, U.S.A

[4] Johan Ingvast. 2006, "*Quadruped Robot Control and Variable Leg Transmissions"*, KTH Industrial Engineering and Management, Stockholm, Sweden.

[5] Tsu-Tian Lee & Ching-Long Shih. 1986, "*A Study of the Gait Control of a Quadruped Walking Vehicle"*, IEEE Journal of Robotics and Automation, Vol. RQ-2, No. 2, June 1986

[6] J. Z. Kolter, M. A. Rodgers & Y. N. Andrew. 2008, "*A Control Architecture for Quadruped Locomotion over Rough Terrain"*, Stanford University, Stanford

[7] R. B. McGhee & A. A. Frank. 1968, "*On The Stability Properties of Quadruped Creeping Gaits"*, J. Math. Biosciences

[8] J. Iovine. 2004, *PIC Microcontroller Project Book,* McGraw-Hill, U.S.A

[9] M. Predko. 2003, *Programming Robot Controllers,* McGraw-Hill, U.S.A

[10] M. A. Mazidi, R. D. Mckinlay & D. Causey. 2008, *PIC Microcontroller and Embedded Systems: Using Assembly and C for PIC18,* Prentice Hall, New Jersey

[11] R. Barnett, L. O'Cull & S. Cox. 2004, *Embedded C Programming and the Microchip PIC,* Thomson Delmar Learning, New York

APPENDICES

APPENDIX 1

Example Crawling Gait Cycle of Quadruped Robot

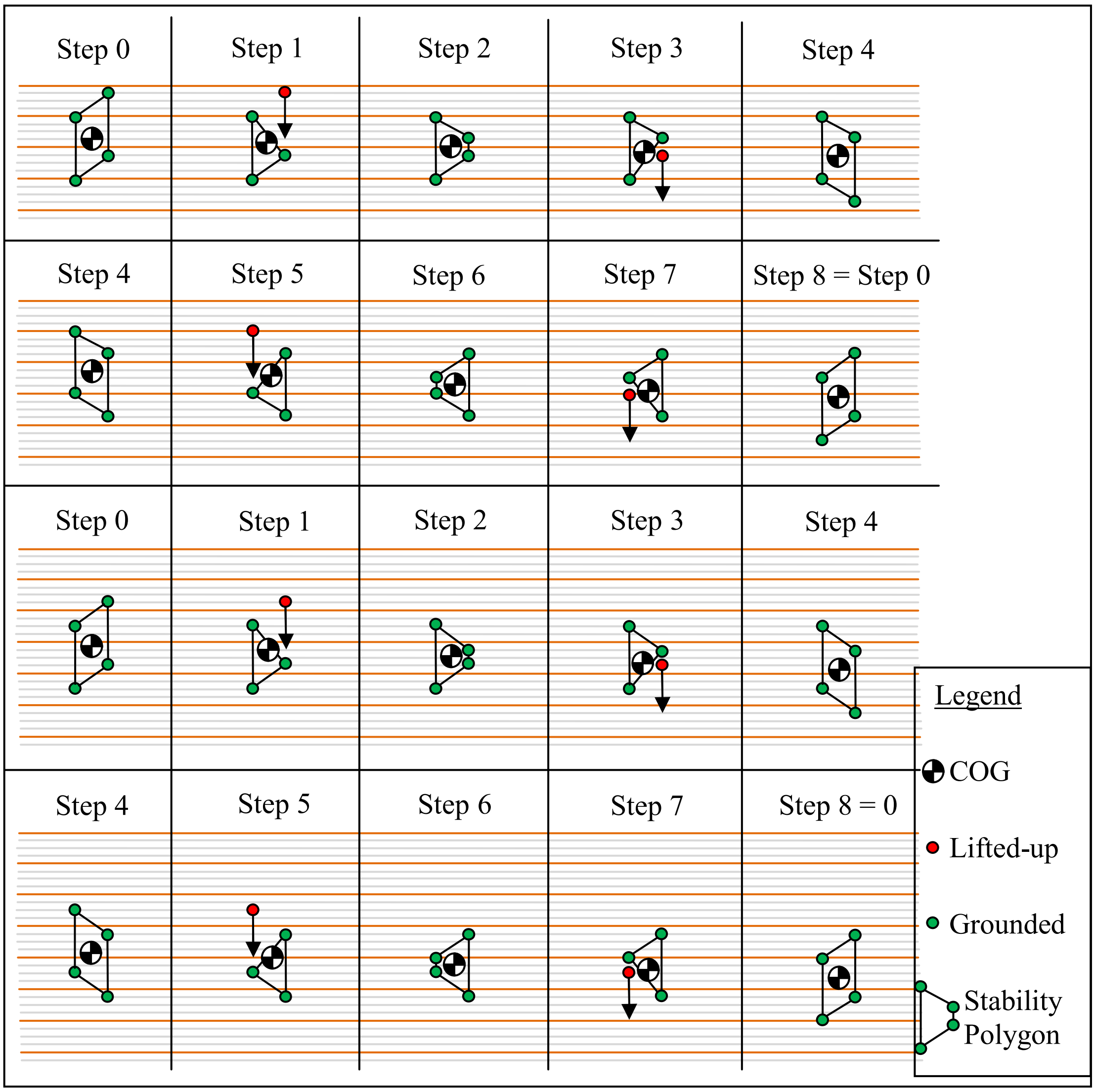
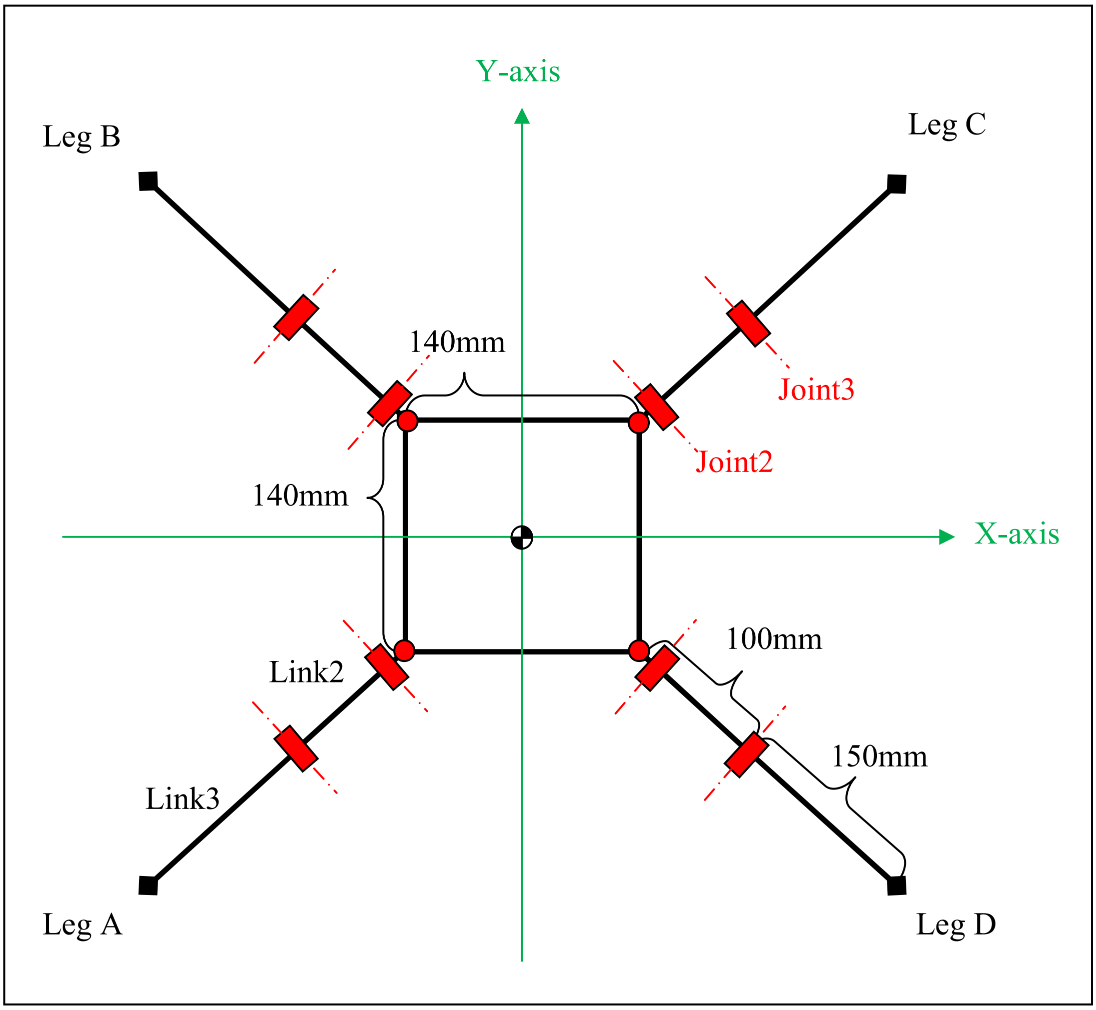


Figure above illustrates the crawling gait sequence of quadruped robot from top view. Taking step 0 as the original state of position with four legs grounded on floor forming a parallelogram, when one leg is lifted up (the red dot represents a lifted leg), a triangular polygon is formed by the remaining legs. These triangular polygon and parallelogram are known as stability polygon. It is noted that the center of gravity is fall within the constraint of stability polygon to avoid tip-over [4]. From step 0 to step 8 is considered as one cycle where one leg is allow to lift up once per cycle. Step 8 has the same posture with step 0 but at a farther distance from the original position. At this point, the quadruped robot can be reckoned as reaching the end of a crawling gait cycle and a new cycle will be started.

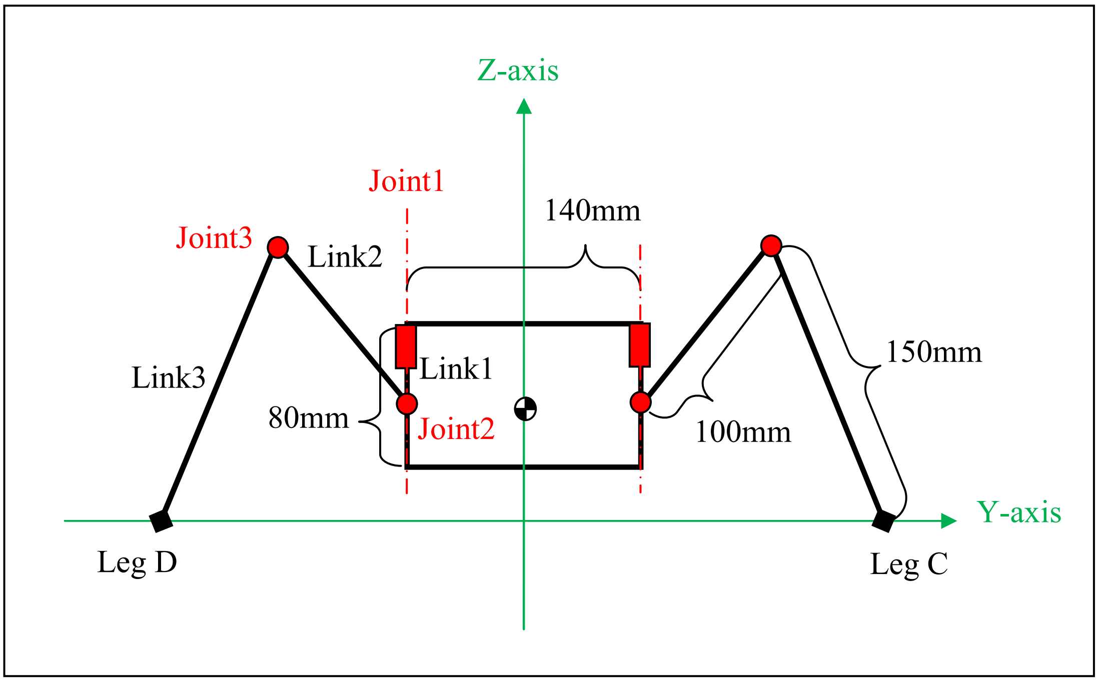
APPENDIX 2

Schematic Design of Quadruped (Top View)



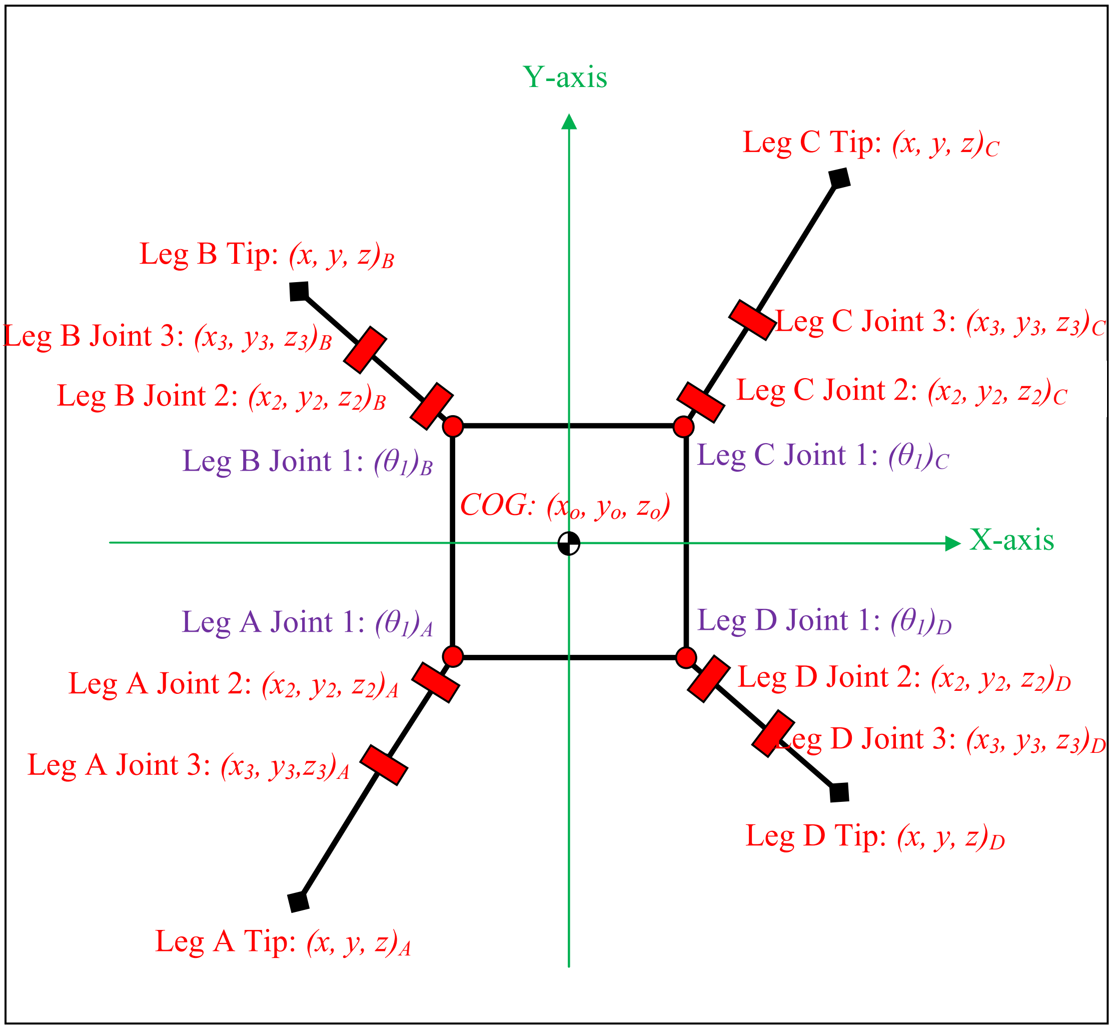
APPENDIX 3

Schematic Design of Quadruped (Right-hand Side View)



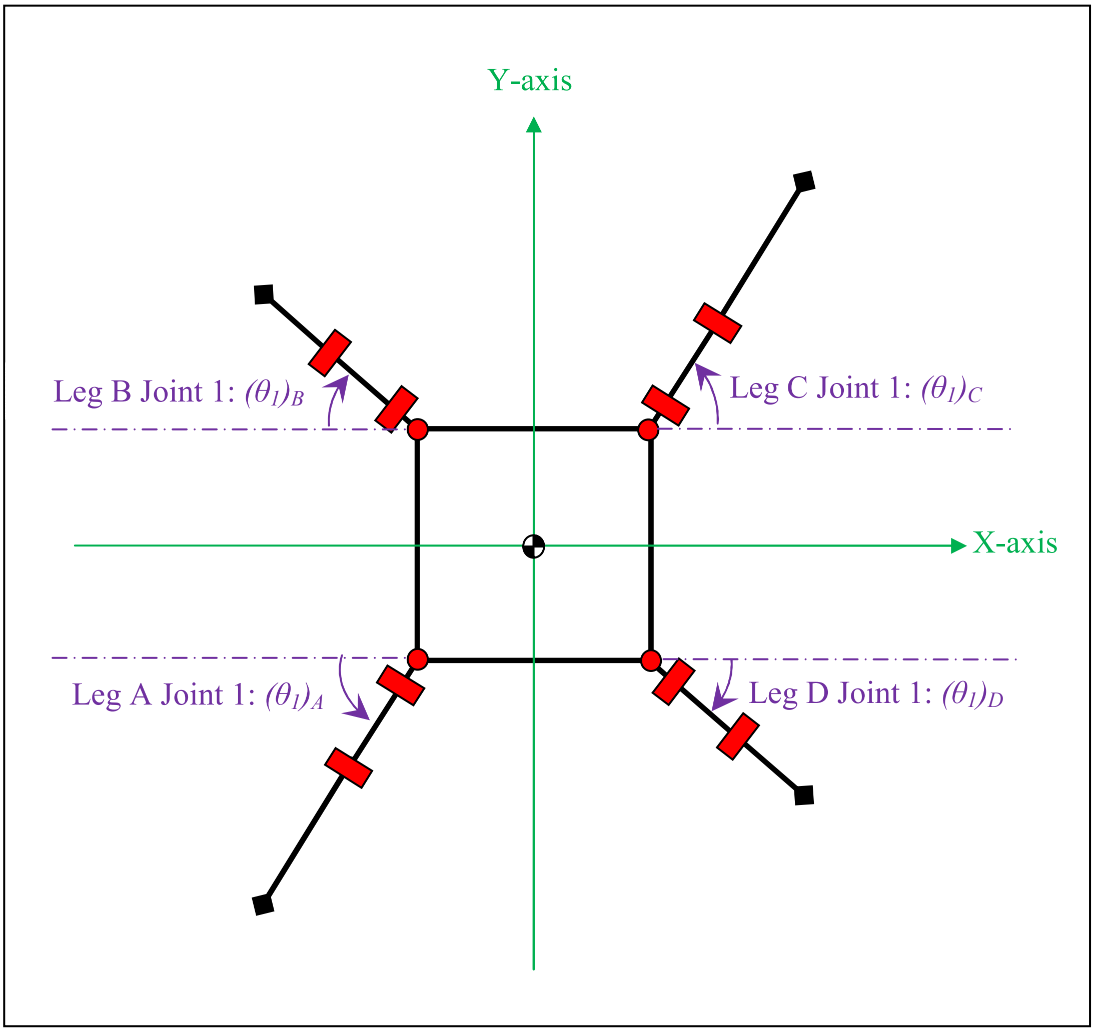
APPENDIX 4

Global Coordinate Notation of Quadruped (Top View)


Y-axis
Leg C Tip: $(x, y, z)_C$
Leg B Tip: $(x, y, z)_B$
Leg C Joint 3: $(x_3, y_3, z_3)_C$
Leg B Joint 3: $(x_3, y_3, z_3)_B$
Leg B Joint 2: $(x_2, y_2, z_2)_B$
Leg C Joint 2: $(x_2, y_2, z_2)_C$
Leg B Joint 1: $(\theta_1)_B$
Leg C Joint 1: $(\theta_1)_C$
COG: (x_o, y_o, z_o)
X-axis
Leg A Joint 1: $(\theta_1)_A$
Leg D Joint 1: $(\theta_1)_D$
Leg A Joint 2: $(x_2, y_2, z_2)_A$
Leg D Joint 2: $(x_2, y_2, z_2)_D$
Leg D Joint 3: $(x_3, y_3, z_3)_D$
Leg A Joint 3: $(x_3, y_3,z_3)_A$
Leg D Tip: $(x, y, z)_D$
Leg A Tip: $(x, y, z)_A$

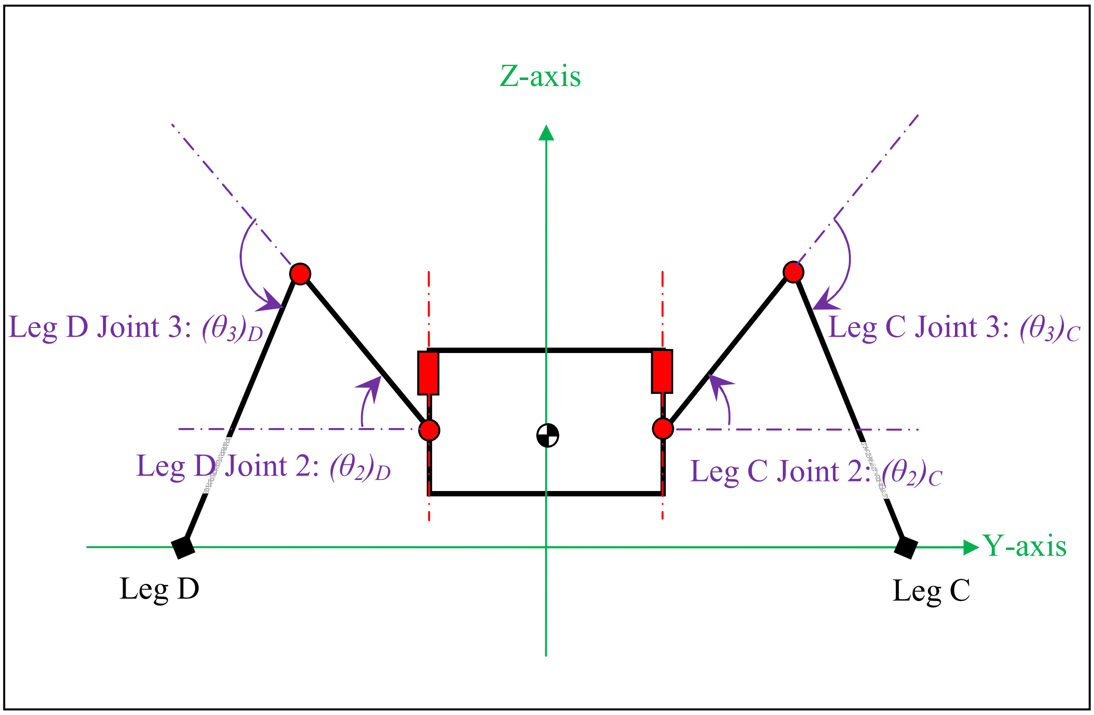
APPENDIX 5

Joint Angle of Quadruped (Top View)


Y-axis
Leg B Joint 1: $(\theta_1)_B$
Leg C Joint 1: $(\theta_1)_C$
X-axis
Leg A Joint 1: $(\theta_1)_A$
Leg D Joint 1: $(\theta_1)_D$

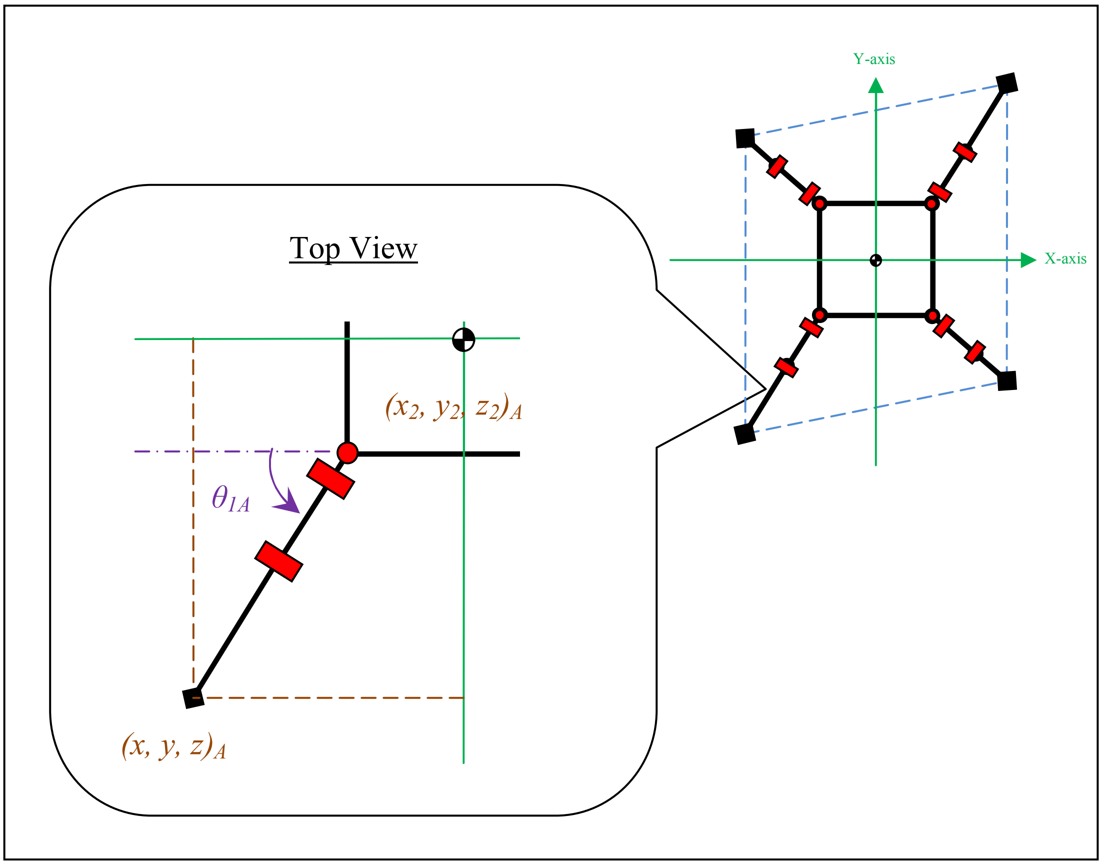
APPENDIX 6

Joint Angle of Quadruped (Right-hand Side View)


Z-axis
Leg D Joint 3: $(\theta_3)_D$
Leg C Joint 3: $(\theta_3)_C$
Leg D Joint 2: $(\theta_2)_D$
Leg C Joint 2: $(\theta_2)_C$
Y-axis
Leg D
Leg C

APPENDIX 7

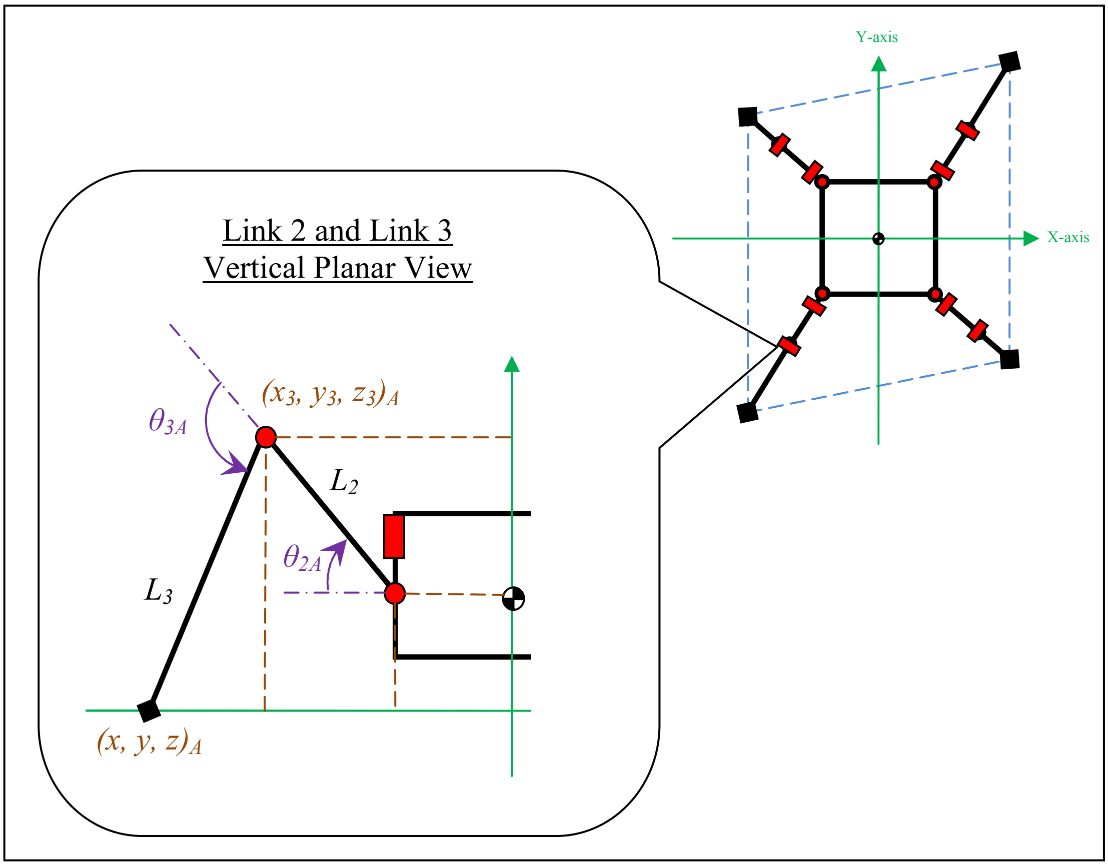
Example of Joint Angle and Coordinates Calculation



Taking leg A as example, the joint angle of joint 1, θ_{1A} can be obtained by using *Trigonometry* provided the coordinate of joint 2 and foot tip are known. Note that purple line is a reference axis where the arrow starts from. Joint 1 angle, θ_{1A} is positive if foot tip stays below the reference axis, whereas it goes inversely if foot tip stays above the reference axis.

$$\tan(\theta_{1A}) = \frac{y_{2A} - y_A}{x_{2A} - x_A}$$

$$\therefore \theta_{1A} = \tan^{-1}\left(\frac{y_{2A} - y_A}{x_{2A} - x_A}\right)$$



Imagine link 2 and link 3 of leg A forms a vertical plane where joint 2 angle, θ_{2A} and joint 3 angle, θ_{3A} can be seen clearly from this view. θ_{2A} and θ_{3A} can be obtained by using *Trigonometry* and *Law of Cosines* provided the coordinate of joint 2 and foot tip, length of link 2 (L_2) and 3 (L_3) are known. The calculation of θ_{2A} is shown as below.

Law of Cosines:

$$L_3^2 = L_2^2 + (x_{2A} - x_A)^2 + (y_{2A} - y_A)^2 + (z_2 - z_O)^2 -$$

$$2L_2\sqrt{(x_{2A} - x_A)^2 + (y_{2A} - y_A)^2 + (z_2 - z_O)^2}\cos\Bigg(\theta_{2A} +$$

$$\tan^{-1}\left(\frac{z_2 - z_O}{\sqrt{(x_{2A} - x_A)^2 + (y_{2A} - y_A)^2}}\right)\Bigg)$$

$$\therefore\ \theta_{2A} = \cos^{-1}\left(\frac{L_2^2+(x_{2A}-x_A)^2+(y_{2A}-y_A)^2+(z_2-z_O)^2-L_3^2}{2L_2\sqrt{(x_{2A}-x_A)^2+(y_{2A}-y_A)^2+(z_2-z_O)^2}}\right) -$$

$$\tan^{-1}\left(\frac{z_2-z_O}{\sqrt{(x_{2A}-x_A)^2+(y_{2A}-y_A)^2}}\right)$$

The joint 3 angle, θ_{3A} can be obtained by using the same approach as calculating θ_{2A}. The calculation of θ_{3A} is shown as below.

Law of Cosines:

$$(x_{2A}-x_A)^2+(y_{2A}-y_A)^2+(z_2-z_O)^2 = L_2^2+L_3^2-$$

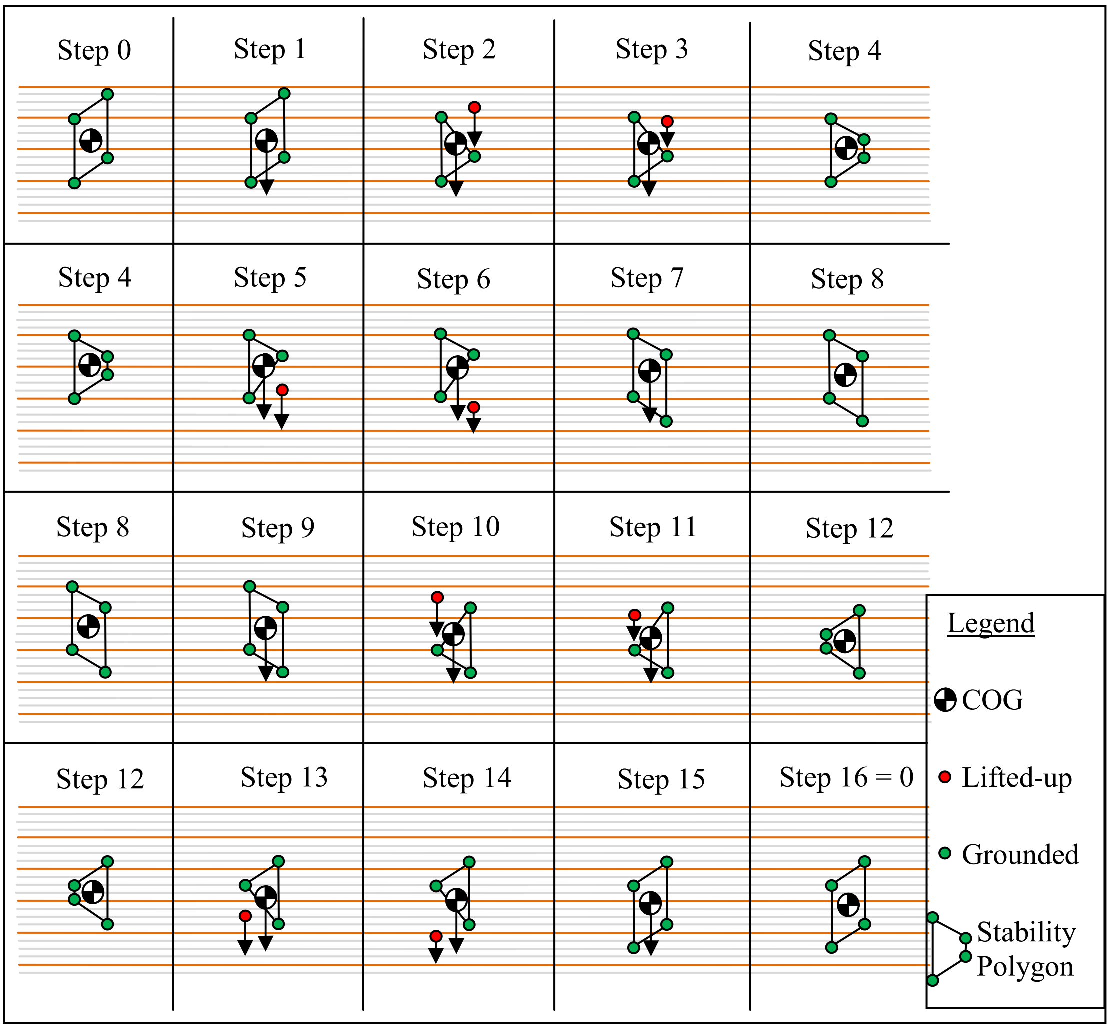
$$2L_2\,L_3\cos(180-\theta_{3A})$$

$$(x_{2A}-x_A)^2+(y_{2A}-y_A)^2+(z_2-z_O)^2 = L_2^2+L_3^2+\ 2L_2\,L_3\cos(\theta_{3A})$$

$$\therefore\ \theta_{3A} = cos^{-1}\left(\frac{(x_{2A}-x_A)^2+(y_{2A}-y_A)^2+(z_2-z_O)^2-L_2^2-L_3^2}{2L_2L_3}\right)$$

APPENDIX 8

Crawling Gait of 16 Steps (Timing Adjusted)



At step 1, all legs remain grounded and push the COG forward. No leg is lifted up at step 1 because lifting one leg up will leave the COG lay right above the boundary line of stability polygon and thus static stability is hardly secured. This problem has been explained in *Section 2.6*. Same approach is used at step 9 as it has the same problem with step 1. Note that the red-colored leg at step 6 is landed at step 7 instead of step 8. This is so because of the stability problem again. If the red-colored leg at step 6 does not land at step 7 and leave only three legs on the ground, the COG will be too close to the boundary line of stability polygon and the static stability margin might turn negative (means tip-over will occur). Therefore, red-colored leg at step 6 will be landed at step 7 ahead of step 8 to avoid tip-over. Step 15 has the same problem as step 7 and thus same approach is used.

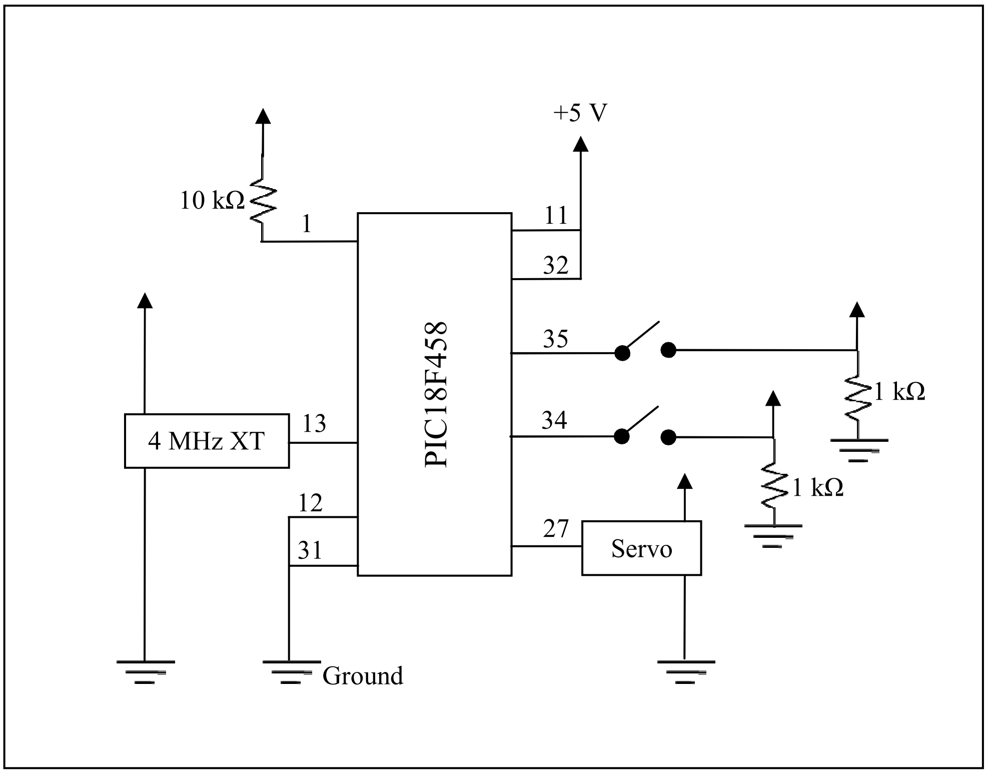
APPENDIX 9

Single Servomotor Program

```
#include            <18F458.h>
#device             adc=8
#use                delay(clock=4000000)
#fuses              NOWDT,WDT128,RC_IO, NOPROTECT, NOOSCSEN, NOBROWNOUT, BORV20,
                    NOPUT, NOCPD, NOSTVREN, NODEBUG, NOLVP, NOWRT, NOWRTD, NOWRTB,
                    NOCPB, NOWRTC, NOEBTR, NOEBTRB
// Global_Variables
int8                selection, duty;
// RTCC_Interrupt_Function
#int_RTCC
RTCC_isr()
{  // reset RTCC timer
   set_rtcc(47535);
   if(selection & 0x02)
   { duty++;
      if(duty>20)
      {  duty = 20;        }
   }
   else if(selection & 0x04)
   {  duty--;
      if(duty<10)
      {  duty = 10;        }
   }
   else
   {  duty = 15;           }
   // Output logic high for 1 ms
   output_high(PIN_D4);
   delay_ms(1);
   // If duty is 15, output logic high for 0.5 ms more
   switch(duty)
   { case  10:   break;
      case  11:   delay_us(100); break;
      case  12:   delay_us(200); break;
      case  13:   delay_us(300); break;
      case  14:   delay_us(400); break;
      case  15:   delay_us(500); break;
      case  16:   delay_us(600); break;
      case  17:   delay_us(700); break;
      case  18:   delay_us(800); break;
      case  19:   delay_us(900); break;
      case  20:   delay_ms(1); break;
   }
   // Output logic low
   output_low(PIN_D4);
}
void main()
{  // Setting
   setup_adc_ports(NO_ANALOGS);
   setup_adc(ADC_OFF);
   setup_psp(PSP_DISABLED);
   setup_spi(FALSE);
   setup_wdt(WDT_OFF);
   setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
   setup_timer_1(T1_DISABLED);
   setup_timer_2(T2_DISABLED,0,1);
   setup_timer_3(T3_DISABLED|T3_DIV_BY_1);
   setup_comparator(NC_NC_NC_NC);
   enable_interrupts(INT_RTCC);
   enable_interrupts(GLOBAL);

   // Loop_Forever
   while(1)
   {
      selection = input_B();
   }
}
```

APPENDIX 10

Single Servomotor Wiring Diagram



APPENDIX 11

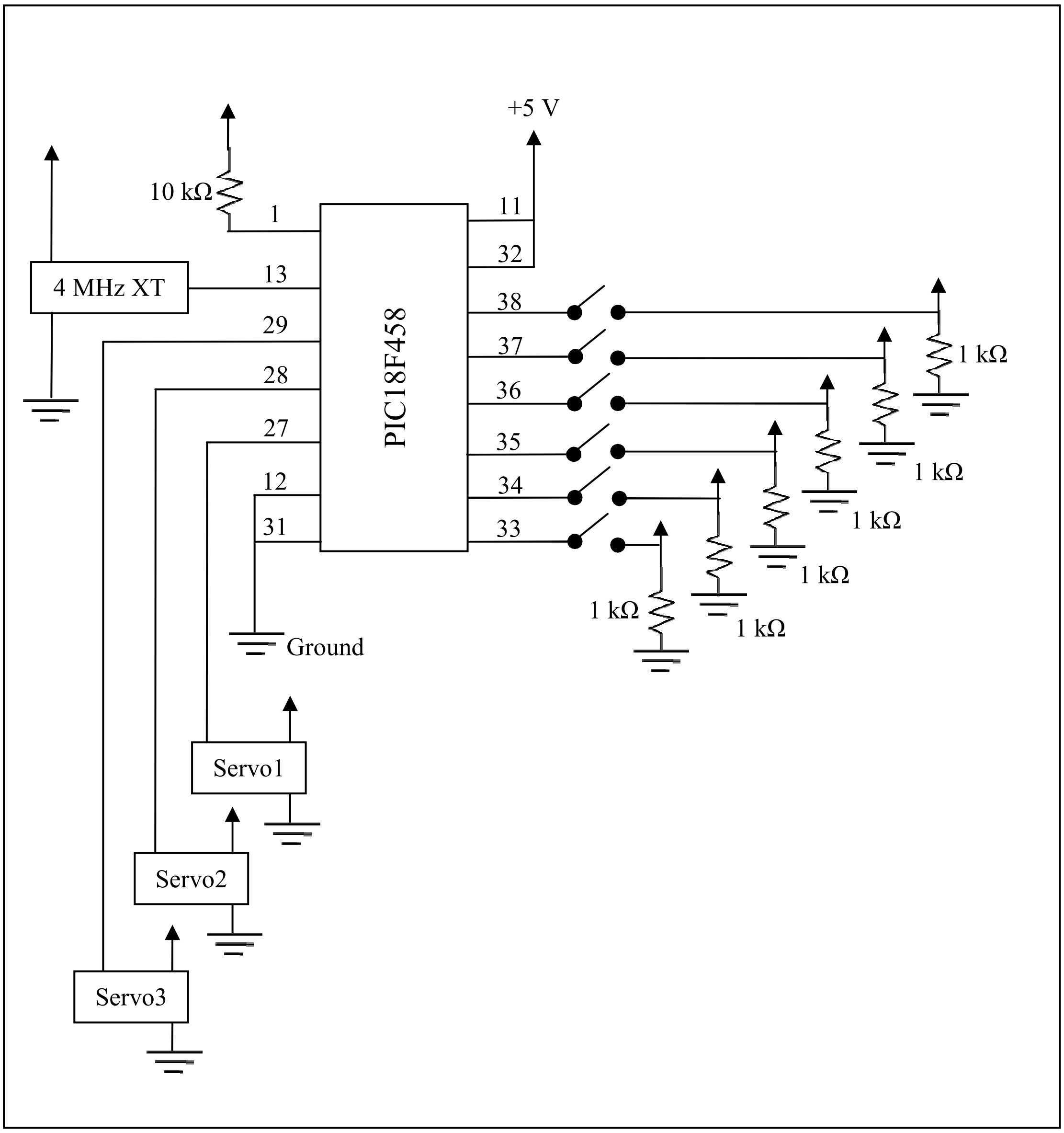
Three Servomotors Program

```
#include            <18F458.h>
#device              adc=8
#use                delay(clock=4000000)
#fuses              NOWDT,WDT128,RC_IO, NOPROTECT, NOOSCSEN, NOBROWNOUT, BORV20,
                    NOPUT, NOCPD, NOSTVREN, NODEBUG, NOLVP, NOWRT, NOWRTD, NOWRTB,
                    NOCPB, NOWRTC, NOEBTR, NOEBTRB
// Global_Variables
int8                selection, duty1, duty2, duty3;

// RTCC_Interrupt_Function
#int_RTCC
RTCC_isr()
{ // reset RTCC timer
  set_rtcc(47535);
  // Motor1 CW
  if(selection & 0x01)
  { if(duty1<10)
    {  duty1 = 15; }
    duty1++;
    if(duty1>20)
    {  duty1 = 20; }
  }
  // Motor1 CCW
  else if(selection & 0x02)
  {
    if(duty1<10)
    {  duty1 = 15; }
    duty1--;
    if(duty1<10)
    {  duty1 = 10; }
  }
  else
  {  duty = 15;          }
  // Output logic high for 1 ms
  output_high(PIN_D4);
  delay_ms(1)
  switch(duty1)
  { case  10:   break;
    case  11:   delay_us(100); break;
    case  12:   delay_us(200); break;
    case  13:   delay_us(300); break;
    case  14:   delay_us(400); break;
    case  15:   delay_us(500); break;
    case  16:   delay_us(600); break;
    case  17:   delay_us(700); break;
    case  18:   delay_us(800); break;
    case  19:   delay_us(900); break;
    case  20:   delay_ms(1); break;
  }
  output_low(PIN_D4);
}
// TIMER1_Interrupt_Function
#int_TIMER1
TIMER1_isr()
{   set_timer1(47535);
  // Motor2 CW
  if(selection & 0x04)
  {
    if(duty2<10)
    {  duty2 = 15; }

    duty2++;

    if(duty2>20)
    {  duty2 = 20; }
  }
  //Motor2 CCW
  else if(selection & 0x08)
  {  if(duty2<10)
    {  duty2 = 15; }
    duty2--;
    if(duty2<10)
    {  duty2 = 10; }
  }
  else
  {  duty2 = 15; }
  output_high(PIN_D5);
  delay_ms(1);

  switch(duty2)
  {
    case  10:   break;
    case  11:   delay_us(100); break;
    case  12:   delay_us(200); break;
    case  13:   delay_us(300); break;
    case  14:   delay_us(400); break;
    case  15:   delay_us(500); break;
    case  16:   delay_us(600); break;
    case  17:   delay_us(700); break;
    case  18:   delay_us(800); break;
    case  19:   delay_us(900); break;
    case  20:   delay_ms(1); break;
  }
  output_low(PIN_D5);
}

// TIMER3_Interrupt_Function
#int_TIMER3
TIMER3_isr()
{ set_timer3(47535);

  // Motor3 CW
  if(selection & 0x10)
  {  if(duty3<10)
    {  duty3 = 15; }
    duty3++;
    if(duty3>20)
    {  duty3 = 20; }
  }

  //Motor3 CCW
  else if(selection & 0x20)
  {  if(duty3<10)
    {  duty3 = 15; }
    duty3--;
    if(duty3<10)
    {  duty3 = 10; }
  }

  else
  {  duty3 = 15; }

  output_high(PIN_D6);
  delay_ms(1);
  switch(duty3)
  { case  10:   break;
    case  11:   delay_us(100); break;
    case  12:   delay_us(200); break;
    case  13:   delay_us(300); break;
    case  14:   delay_us(400); break;
    case  15:   delay_us(500); break;
    case  16:   delay_us(600); break;
    case  17:   delay_us(700); break;
    case  18:   delay_us(800); break;
    case  19:   delay_us(900); break;
    case  20:   delay_ms(1); break;
  }
  output_low(PIN_D6);
}
```

```
void main()
{

  setup_adc_ports(NO_ANALOGS);
  setup_adc(ADC_OFF);
  setup_psp(PSP_DISABLED);
  setup_spi(FALSE);
  setup_wdt(WDT_OFF);
  setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
  setup_timer_2(T2_DISABLED,0,1);
  setup_timer_3(T3_INTERNAL|T3_DIV_BY_1);
  setup_comparator(NC_NC_NC_NC);
  enable_interrupts(INT_RTCC);
  enable_interrupts(INT_TIMER1);
  enable_interrupts(INT_TIMER3);
  enable_interrupts(GLOBAL);

  while(1)
  {
    selection = input_B();
  }
}
```

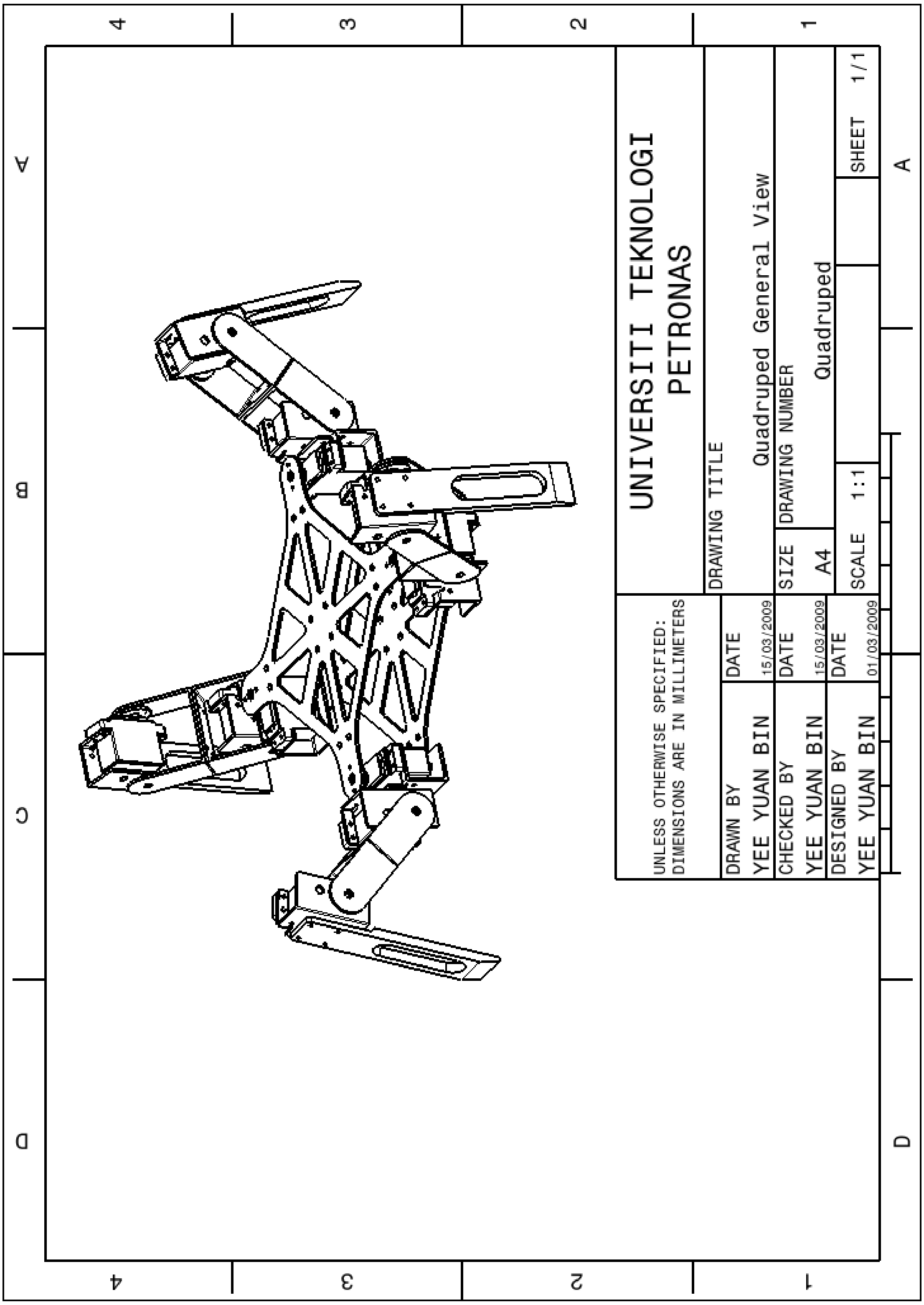
APPENDIX 12

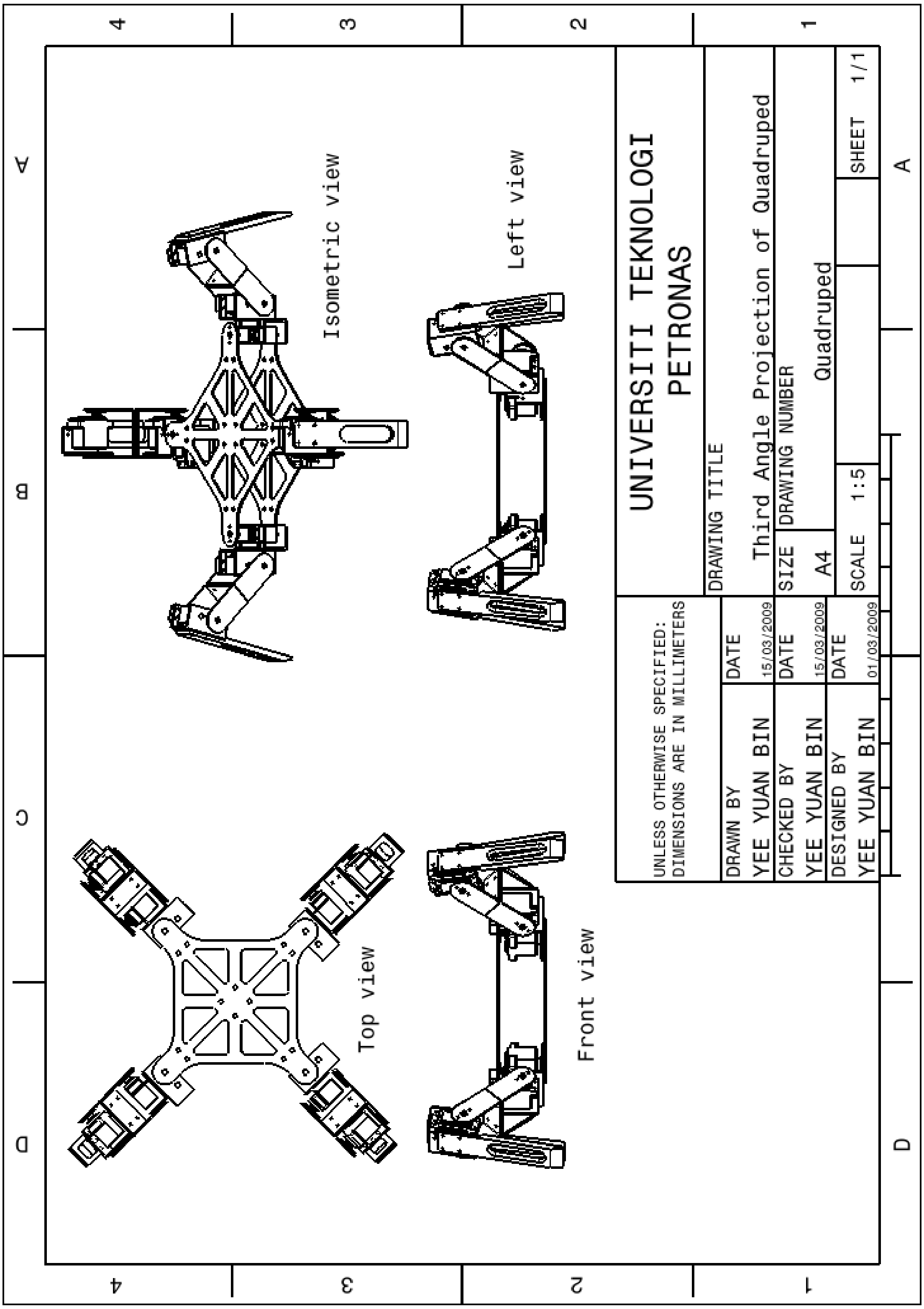
Three Servomotors Wiring Diagram

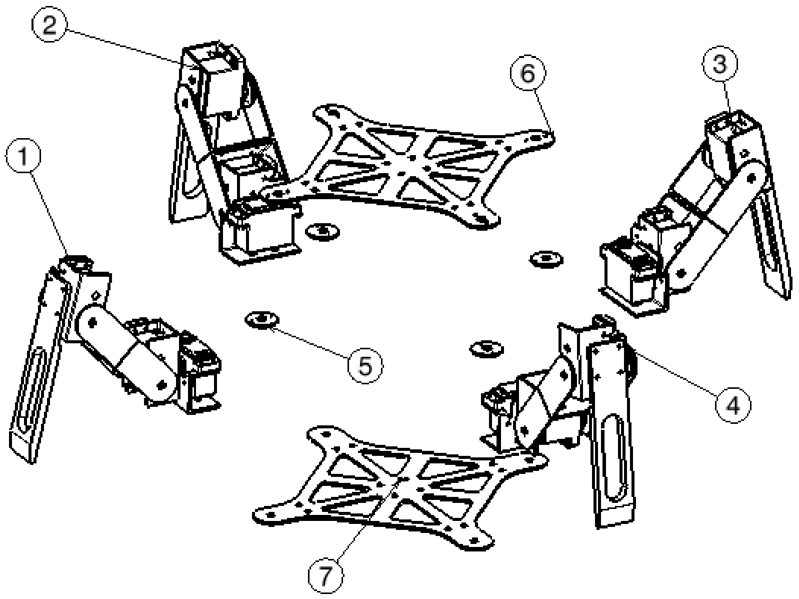


APPENDIX 13

Technical Drawings of Quadruped




Isometric view
Left view
Top view
Front view
UNIVERSITI TEKNOLOGI
PETRONAS
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS
DRAWN BY
YEE YUAN BIN
DATE
15/03/2009
CHECKED BY
YEE YUAN BIN
DATE
15/03/2009
DESIGNED BY
YEE YUAN BIN
DATE
01/03/2009
DRAWING TITLE
Third Angle Projection of Quadruped
SIZE
A4
DRAWING NUMBER
Quadruped
SCALE
1:5
SHEET
1/1
A
B
C
D
1
2
3
4

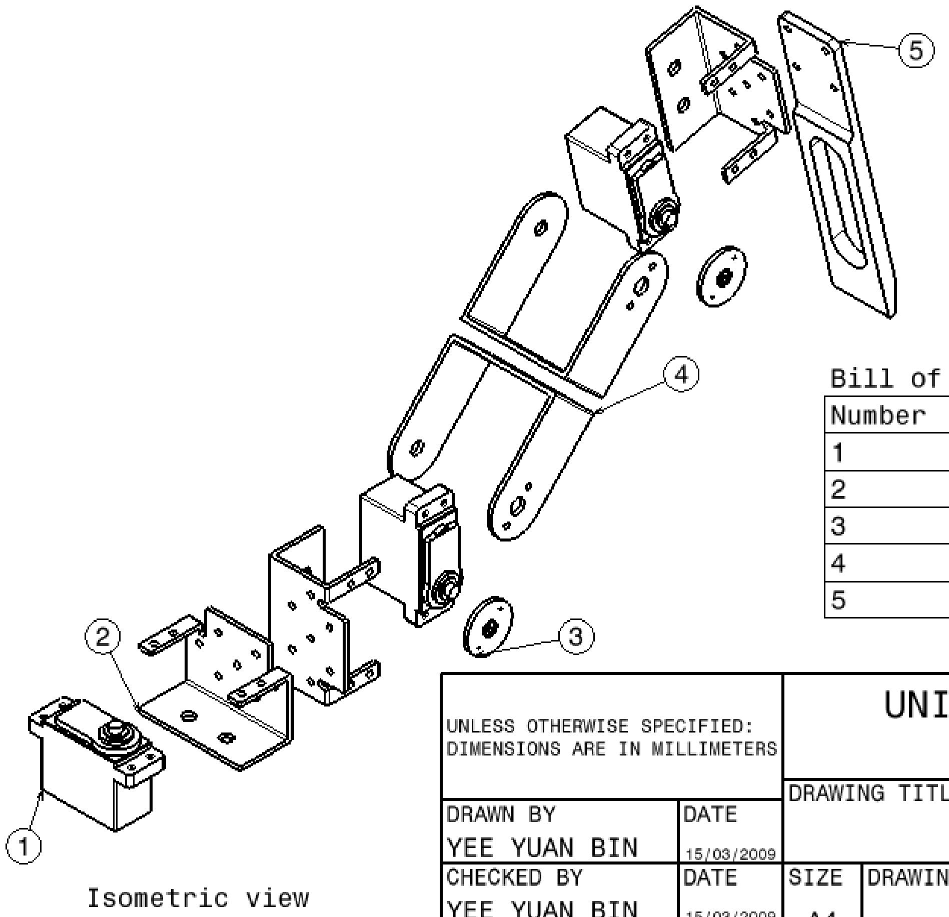


Isometric view

Bill of Material: Quadruped

Number	Part Number	Quantity	Type
1	Leg A	1	Assembly
2	Leg B	1	Assembly
3	Leg C	1	Assembly
4	Leg D	1	Assembly
5	Coupling	4	Part
6	Frame Top	1	Part
7	Frame Bottom	1	Part

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Exploded View of Quadruped	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Exploded-01
DESIGNED BY YEE YUAN BIN	DATE 15/03/2009	SCALE 1:4	SHEET 1/2



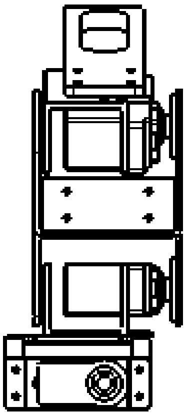
Bill of Material: Leg

Number	Part Number	Quantity	Type
1	Motor	3	Part
2	Bracket J1	3	Part
3	Coupling	2	Part
4	Bracket L2	2	Part
5	Bracket L3	1	Part

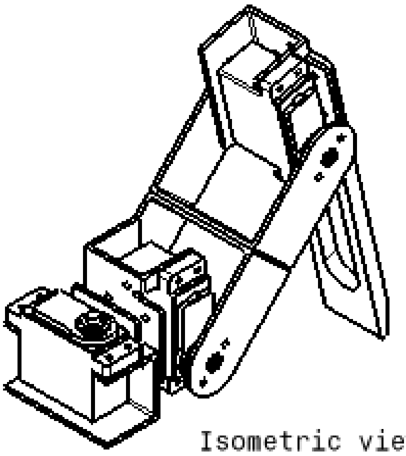
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

UNIVERSITI TEKNOLOGI
PETRONAS

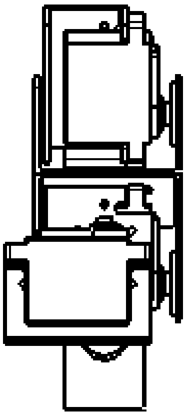
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Exploded View of Leg	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Exploded-02
DESIGNED BY YEE YUAN BIN	DATE 15/03/2009	SCALE 1:2	SHEET 2/2



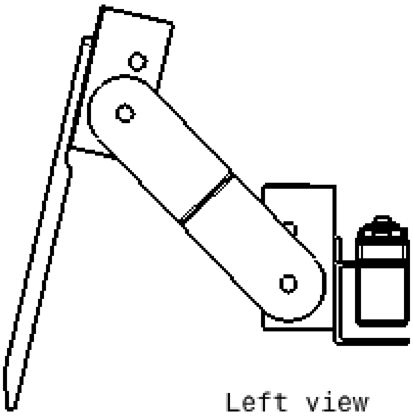
Top view



Isometric view

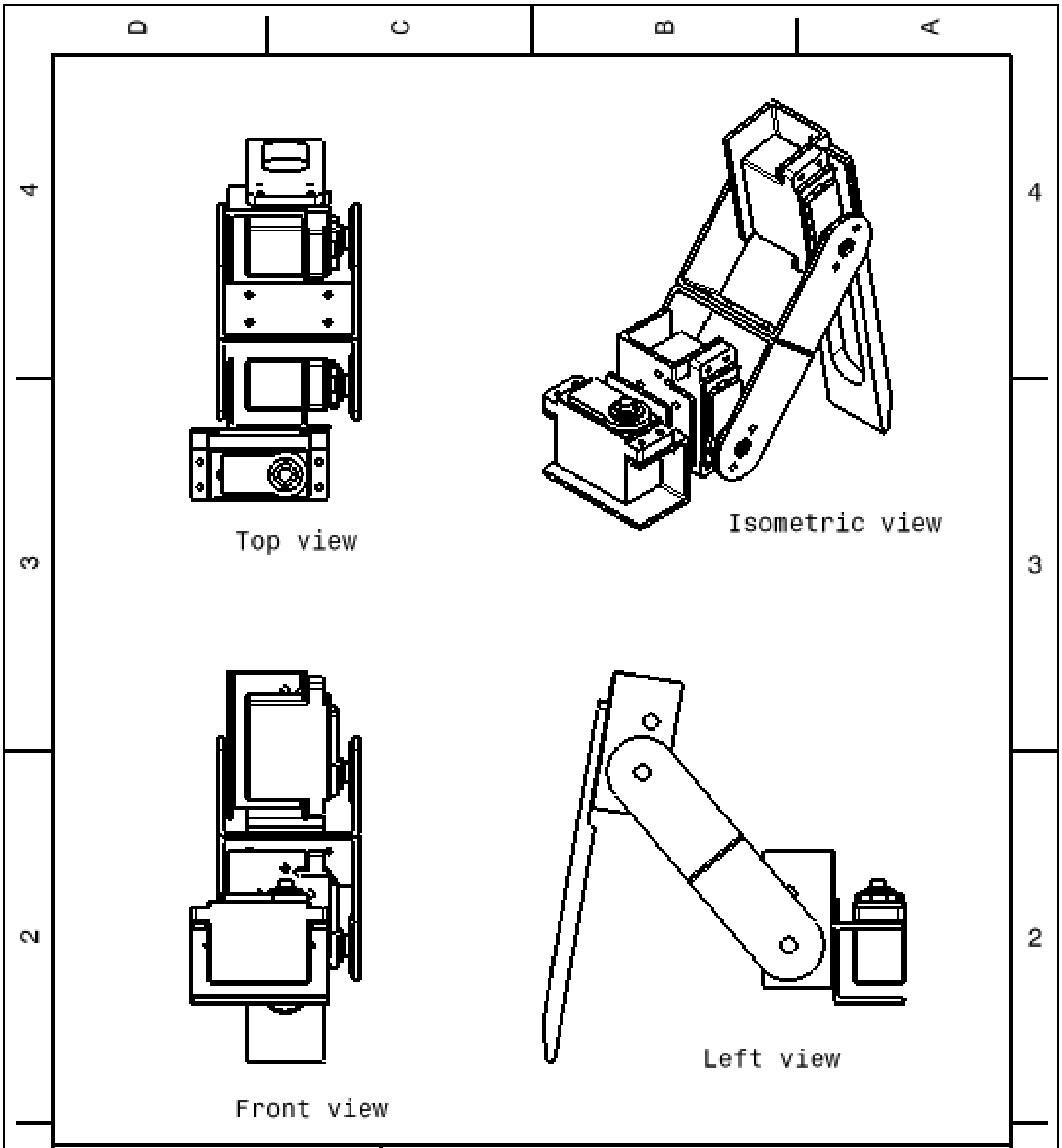


Front view



Left view

UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Third Angle Projection of Leg A	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Leg A
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2	SHEET 1/1

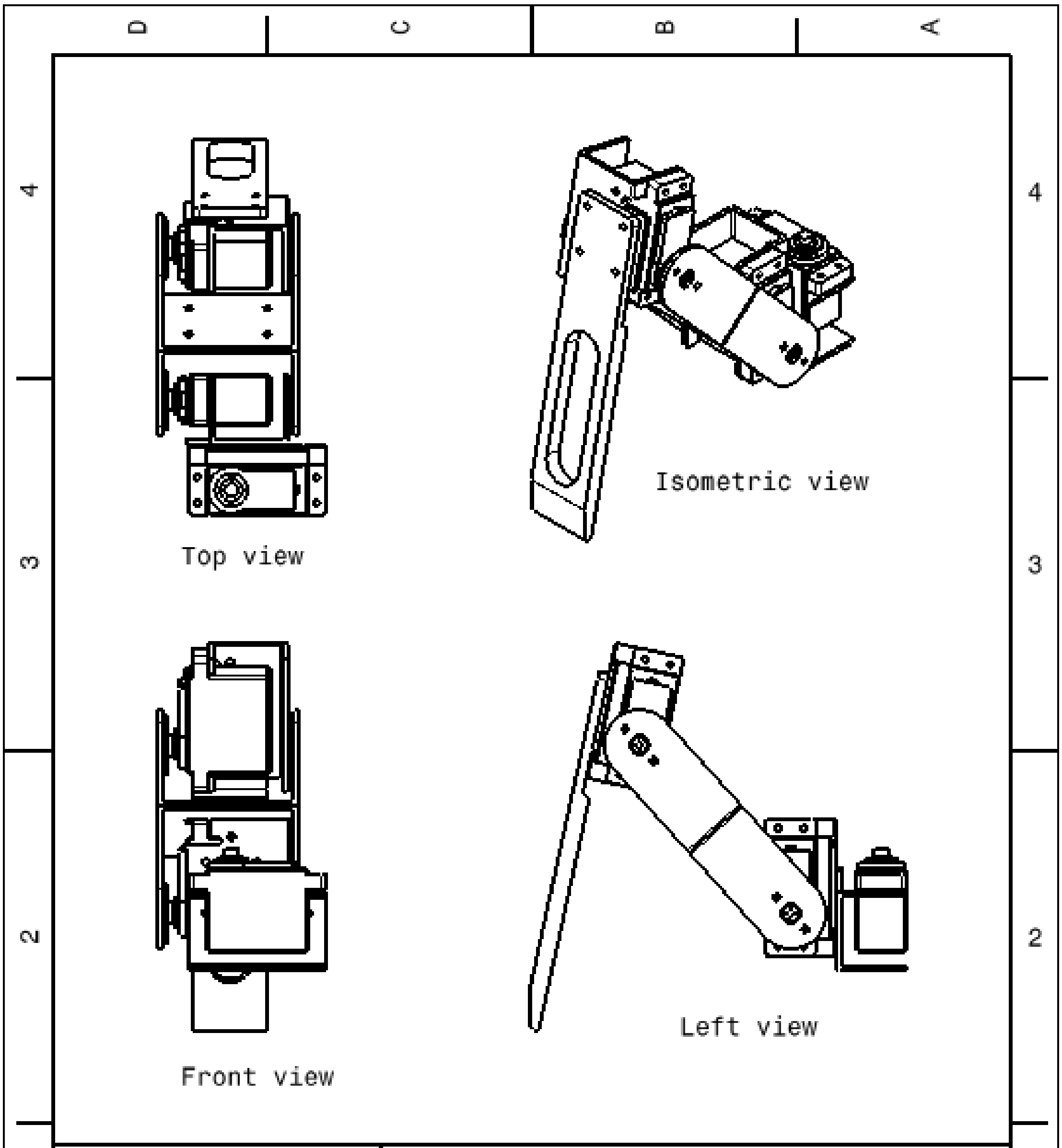


UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

UNIVERSITI TEKNOLOGI
PETRONAS

DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Third Angle Projection of Leg B	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Leg B
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2	SHEET 1/1

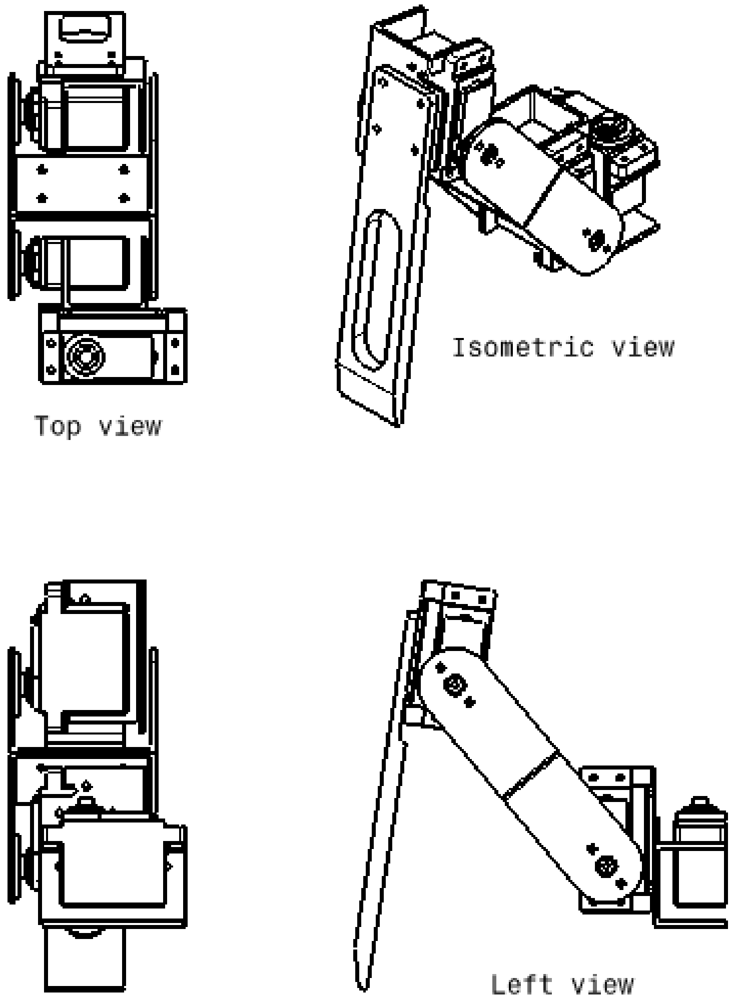
D A



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS			
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Third Angle Projection of Leg C			
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Leg C		
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2			SHEET 1/1

D

A



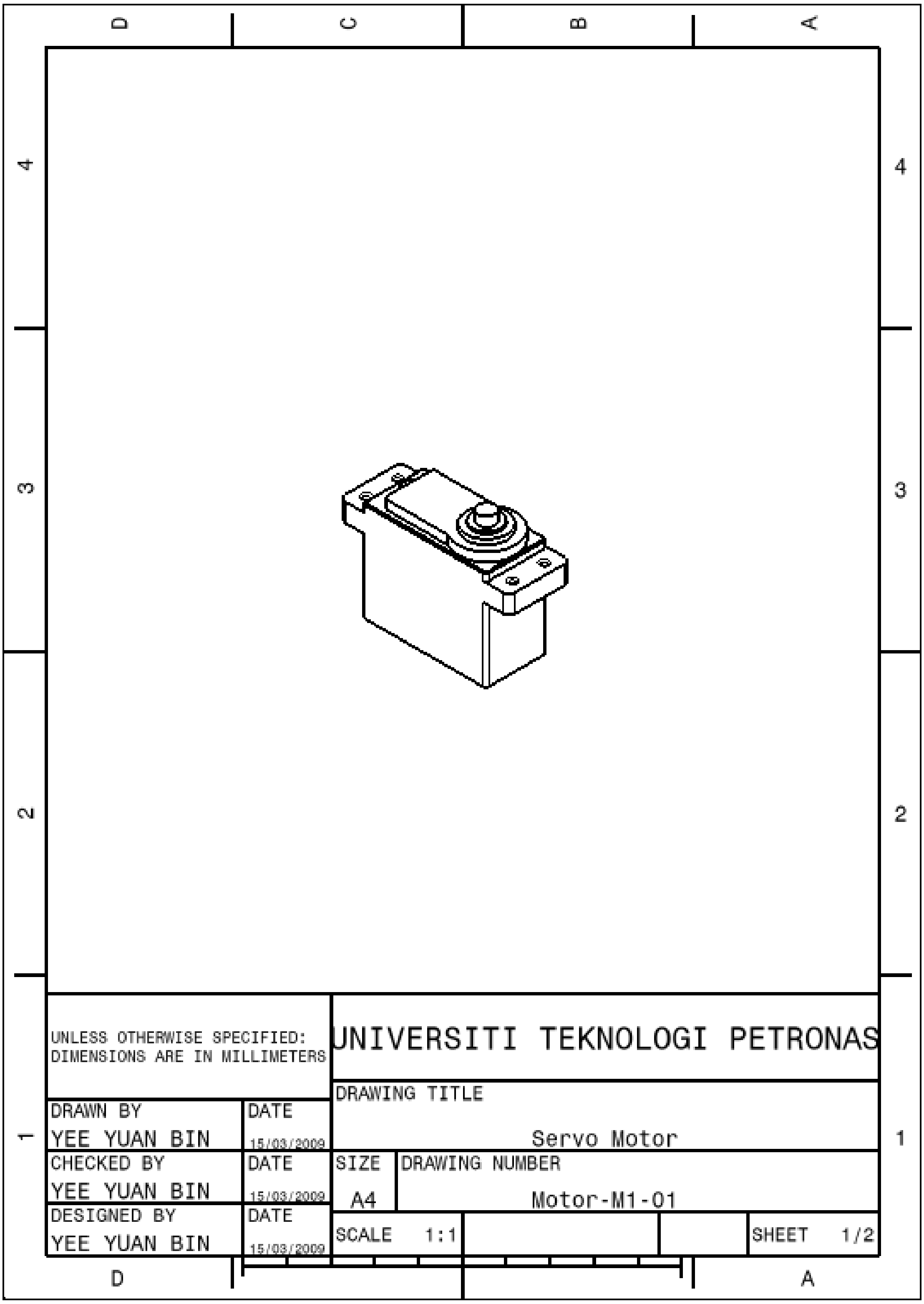
Top view

Isometric view

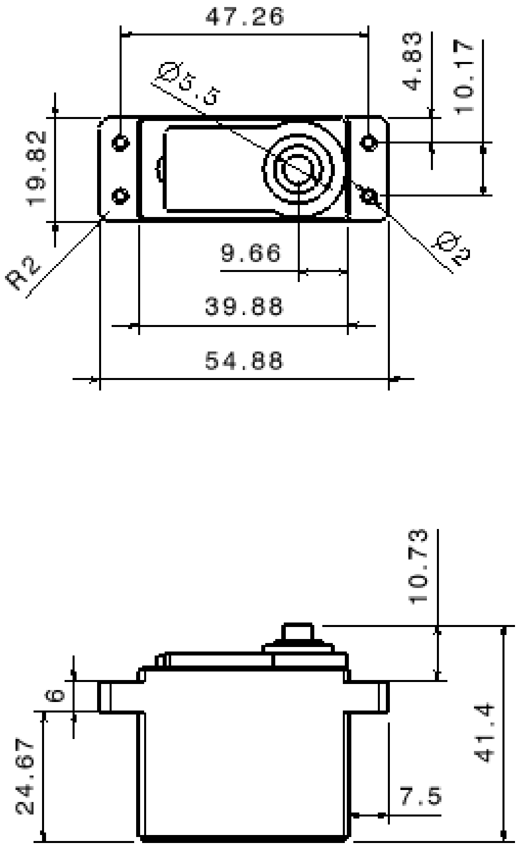
Front view

Left view

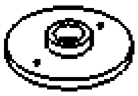
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Third Angle Projection of Leg D	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Leg D
DESIGNED BY YEE YUAN BIN	DATE 15/03/2009	SCALE 1:2	SHEET 1/1



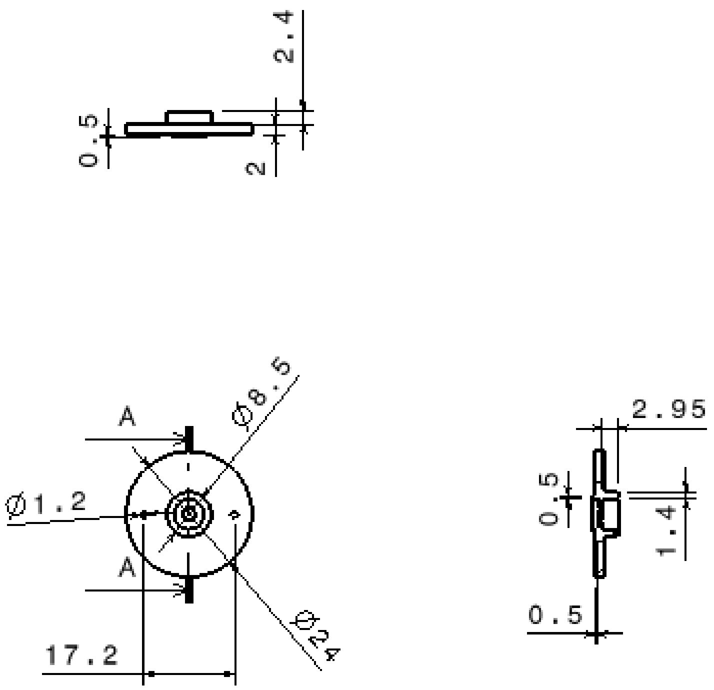
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Servo Motor	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Motor-M1-01
DESIGNED BY YEE YUAN BIN	DATE 15/03/2009	SCALE 1:1	SHEET 1/2



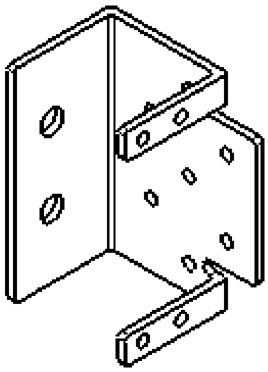
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Servo Motor
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4 | DRAWING NUMBER Motor-M1-02
DESIGNED BY YEE YUAN BIN	DATE 15/03/2009	SCALE 1:1 | SHEET 2/2



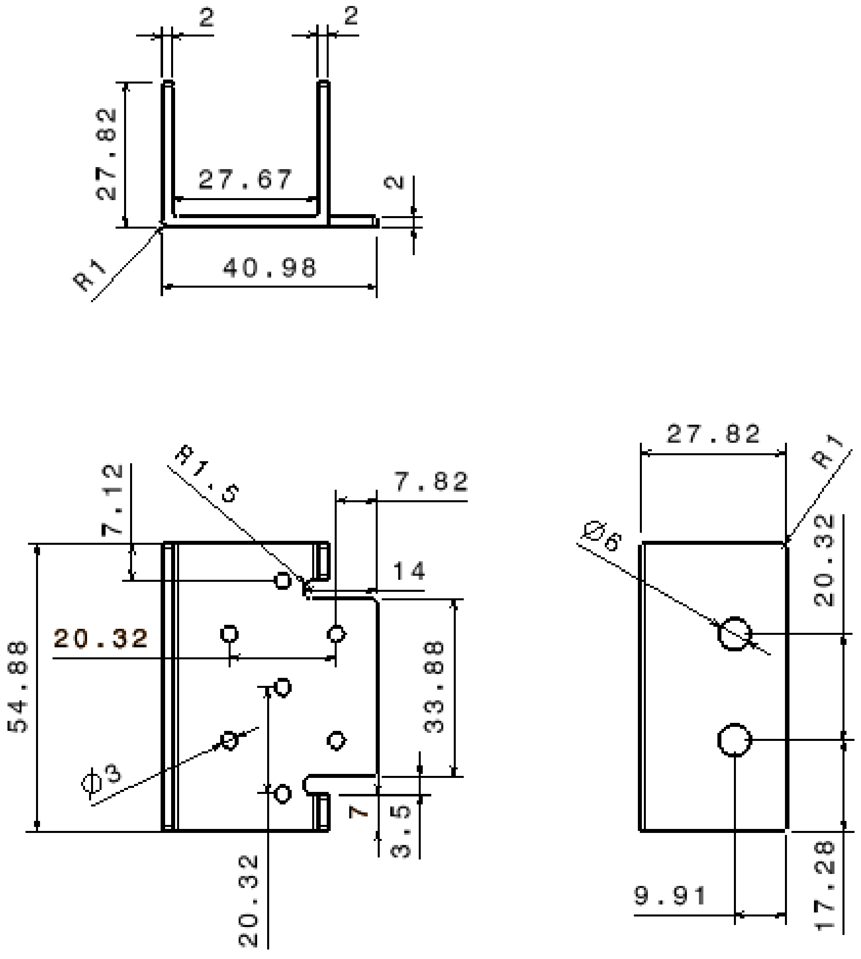
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS			
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Motor Coupling			
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Coupling-C1-01		
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1			SHEET 1/2



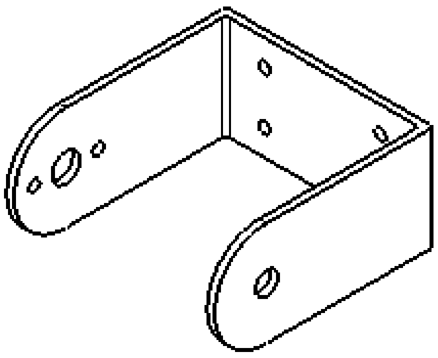
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Motor Coupling	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Coupling-C1-02
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1	SHEET 2/2



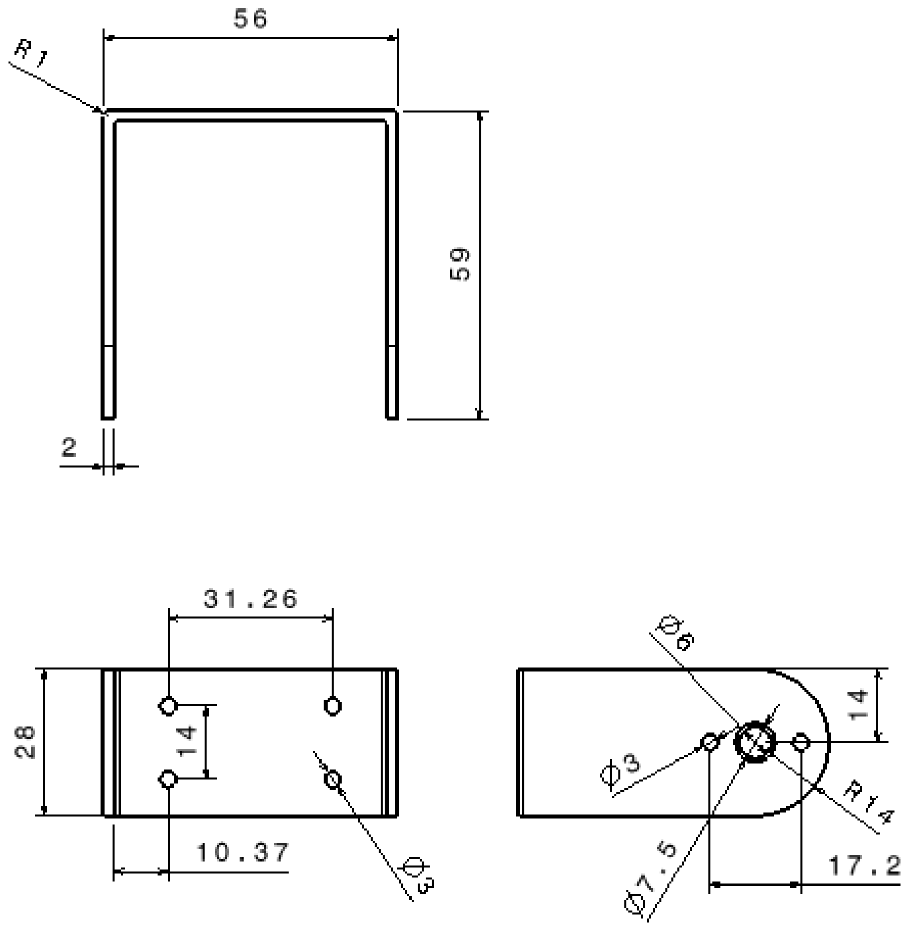
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS		
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Motor Bracket		
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Bracket-J1-01	
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1		SHEET 1/2



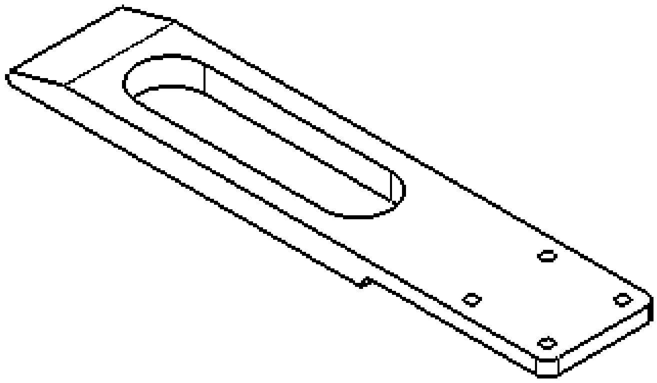
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Motor Bracket	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Bracket-J1-02
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1	SHEET 2/2



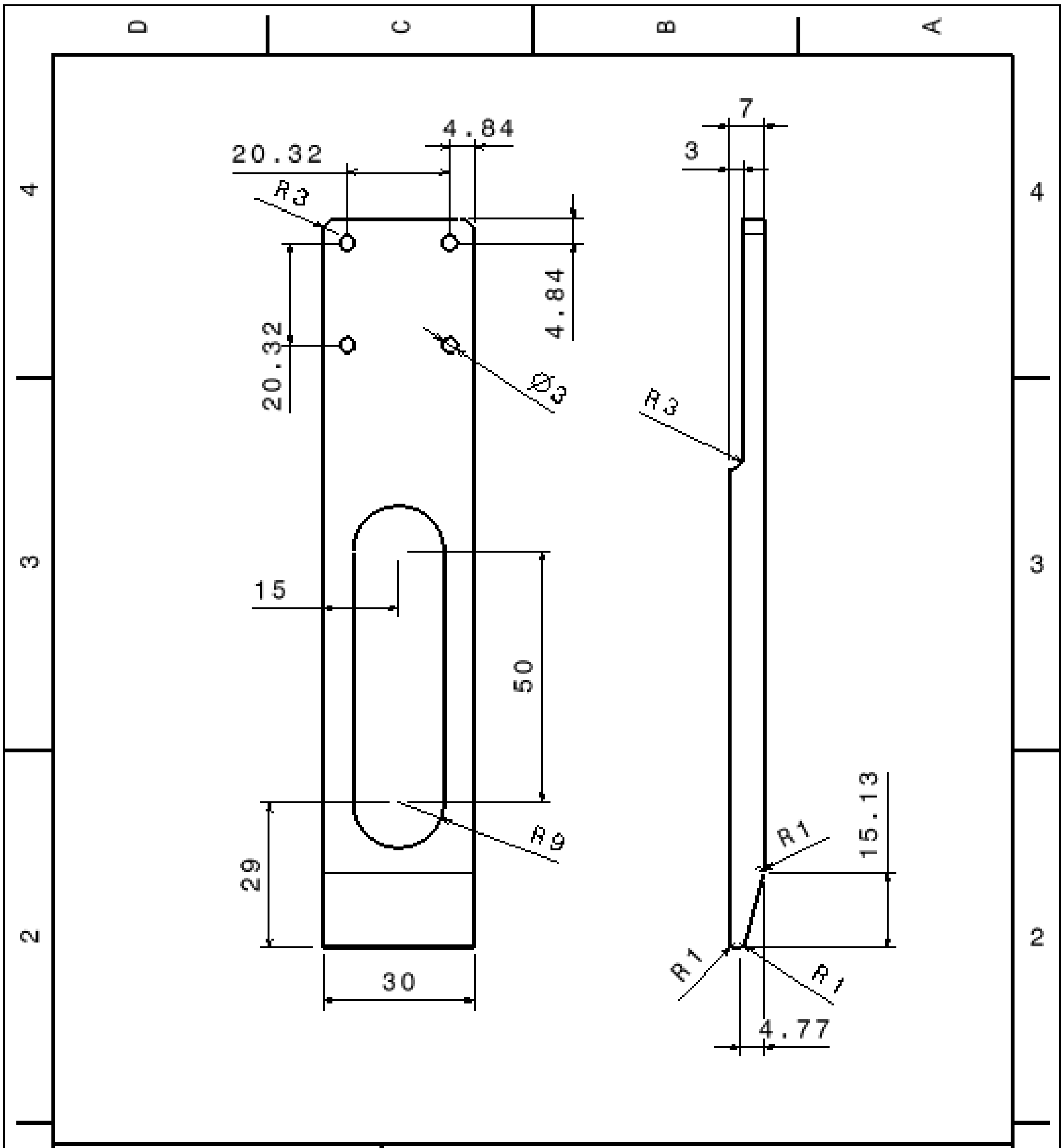
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Linkage Type-I	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Linkage-L2-01
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1	SHEET 1/2



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Linkage Type-I	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Linkage-L2-02
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1	SHEET 2/2

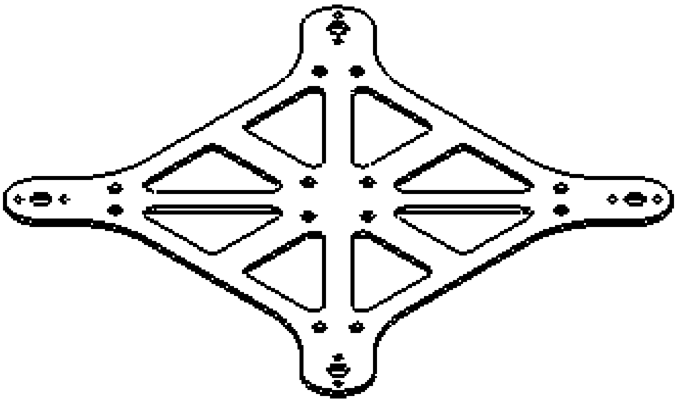


UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS			
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Linkage Type-II			
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Linkage-L3-01		
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1			SHEET 1/2

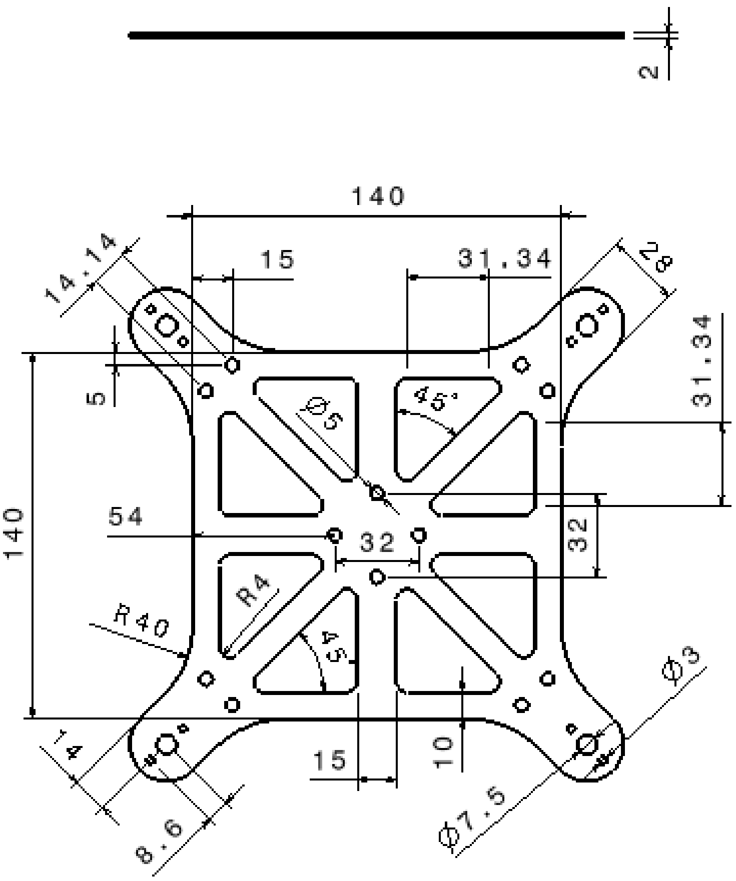


UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Linkage Type-II	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Linkage-L3-02
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:1	SHEET 2/2

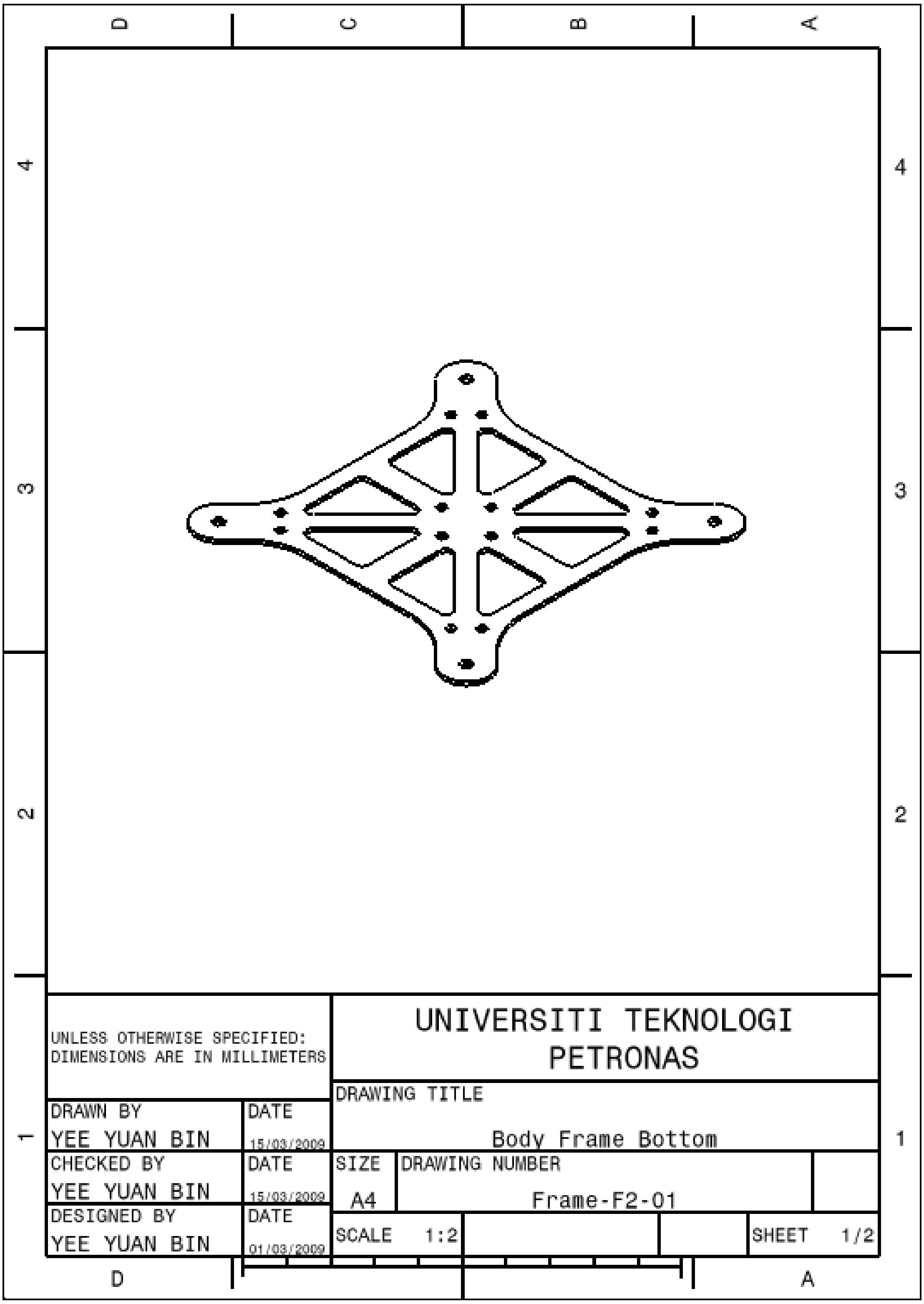
D A



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS			
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Body Frame Top			
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Frame-F1-01		
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2			SHEET 1/2



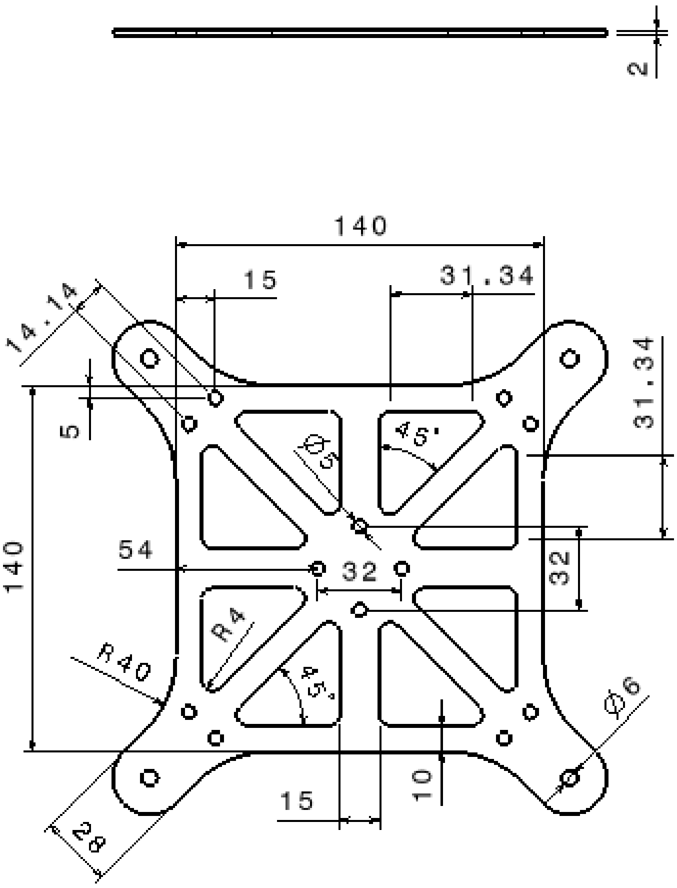
UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS	
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Body Frame Top	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Frame-F1-02
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2	SHEET 2/2



UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN MILLIMETERS

UNIVERSITI TEKNOLOGI
PETRONAS

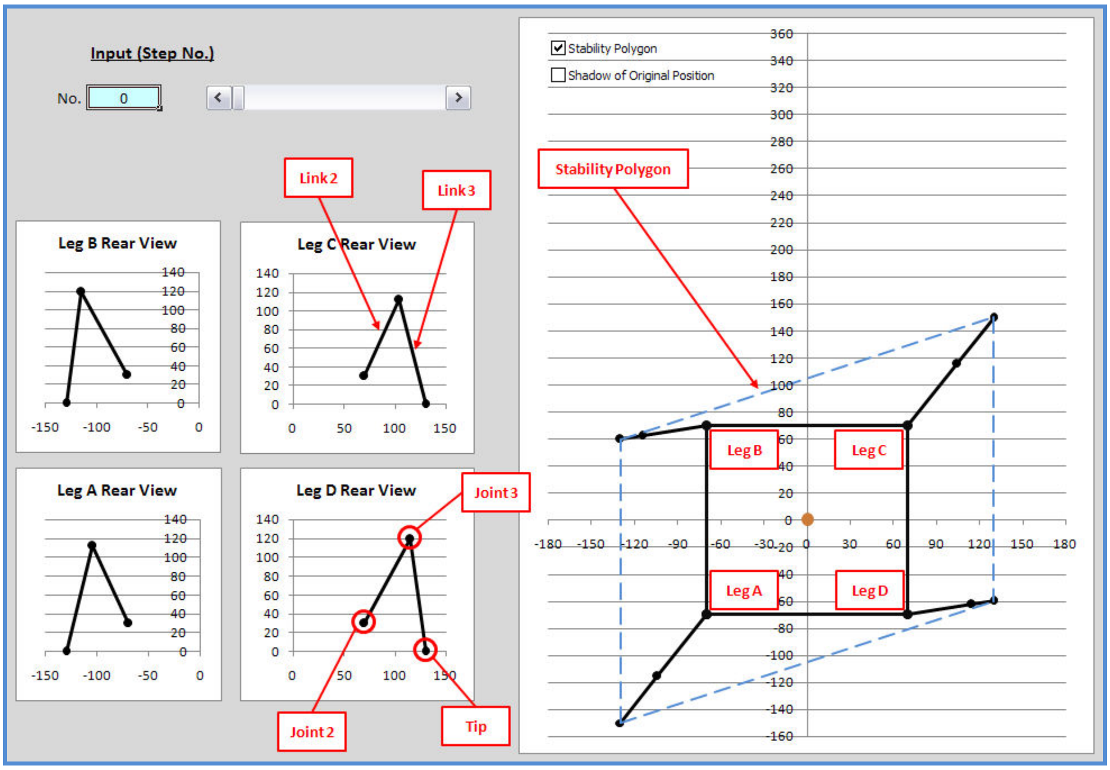
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Body Frame Bottom	
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Frame-F2-01
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2	SHEET 1/2



UNLESS OTHERWISE SPECIFIED: DIMENSIONS ARE IN MILLIMETERS		UNIVERSITI TEKNOLOGI PETRONAS		
DRAWN BY YEE YUAN BIN	DATE 15/03/2009	DRAWING TITLE Body Frame Bottom		
CHECKED BY YEE YUAN BIN	DATE 15/03/2009	SIZE A4	DRAWING NUMBER Frame-F2-02	
DESIGNED BY YEE YUAN BIN	DATE 01/03/2009	SCALE 1:2		SHEET 2/2

APPENDIX 14

Preliminary Gait Simulation
Step 0 (Initial Position)



Input (Step No.)

No. 0

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	70.00	30.00
Joint 3	-114.93	62.51	119.02
Tip	-130.00	60.00	0.00

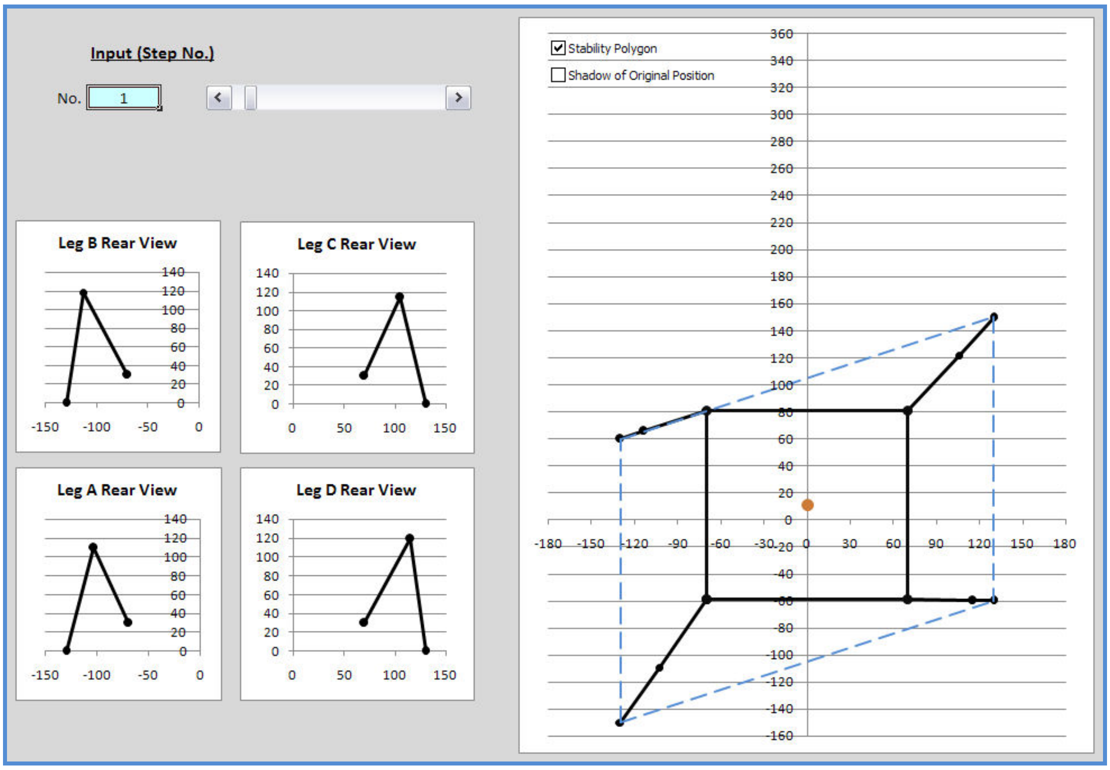
Leg C	x	y	z
Joint 2	70.00	70.00	30.00
Joint 3	104.27	115.70	112.08
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-70.00	30.00
Joint 3	-104.27	-115.70	112.08
Tip	-130.00	-150.00	0.00

Leg D	x	y	z
Joint 2	70.00	-70.00	30.00
Joint 3	114.93	-62.51	119.02
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	53.1	9.5	53.1	9.5
θ2 °	55.2	62.9	55.2	62.9
θ3 °	124.2	145.6	124.2	145.6

Step 1



Input (Step No.)

No. 1

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	81.25	30.00
Joint 3	-113.50	65.84	118.72
Tip	-130.00	60.00	0.00

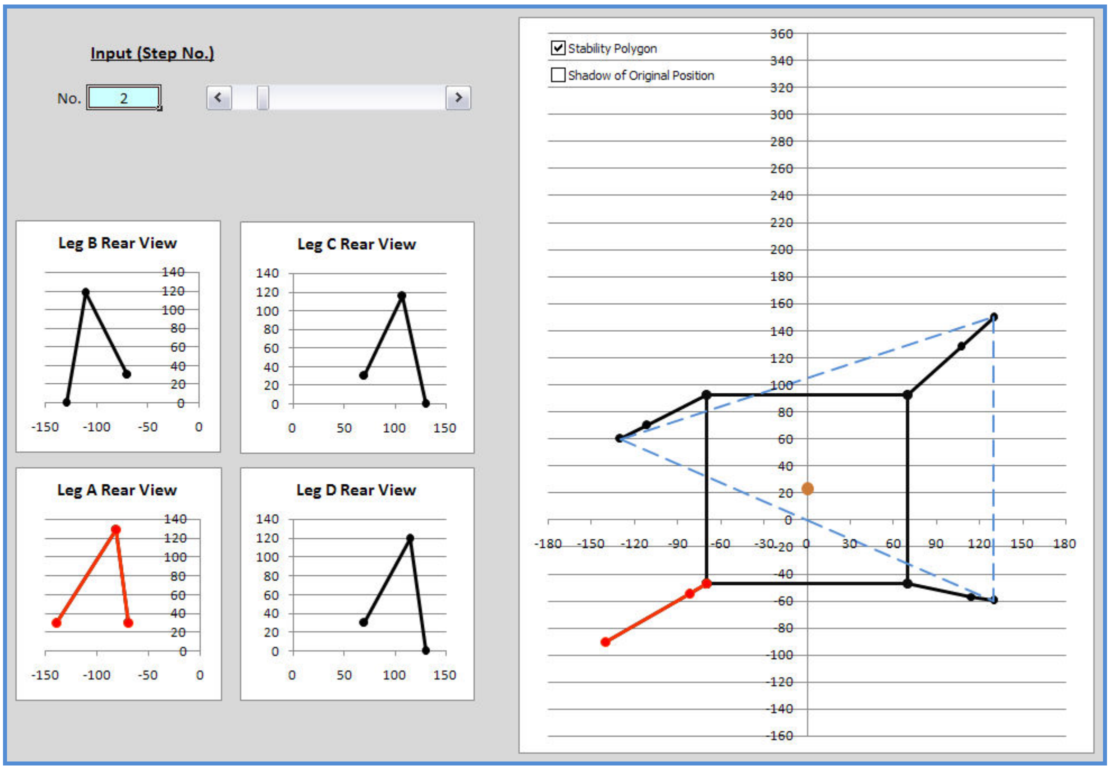
Leg C	x	y	z
Joint 2	70.00	81.25	30.00
Joint 3	105.57	122.01	114.10
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-58.75	30.00
Joint 3	-103.22	-109.27	109.65
Tip	-130.00	-150.00	0.00

Leg D	x	y	z
Joint 2	70.00	-58.75	30.00
Joint 3	115.38	-59.70	119.11
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	56.7	19.5	48.9	-1.2
θ2 °	52.8	62.5	57.2	63.0
θ3 °	118.8	144.1	129.2	146.0

Step 2



Input (Step No.)

No. 2

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	92.50	30.00
Joint 3	-111.52	70.01	118.15
Tip	-130.00	60.00	0.00

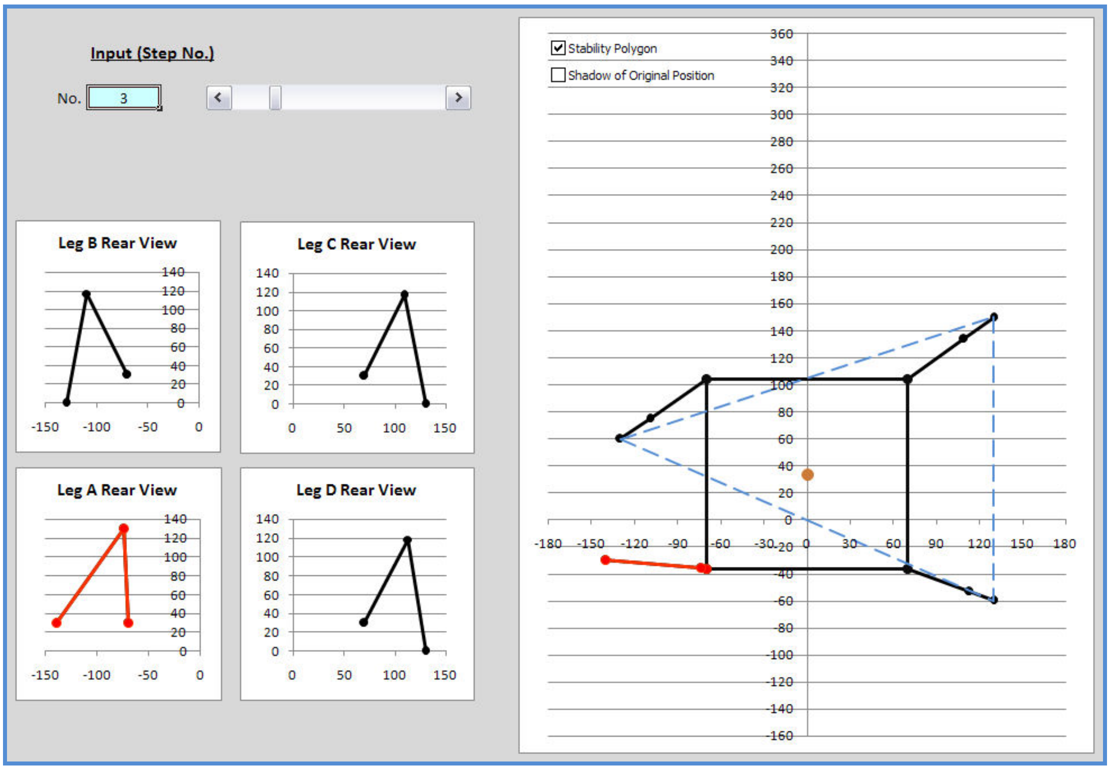
Leg C	x	y	z
Joint 2	70.00	92.50	30.00
Joint 3	107.15	128.10	115.75
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-47.50	30.00
Joint 3	-82.04	-54.81	129.00
Tip	-140.00	-90.00	30.00

Leg D	x	y	z
Joint 2	70.00	-47.50	30.00
Joint 3	114.68	-56.81	118.98
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	31.3	28.4	43.8	-11.8
θ2 °	81.9	61.8	59.0	62.8
θ3 °	137.5	141.7	133.7	145.4

Step 3



Input (Step No.)

No. 3

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	103.75	30.00
Joint 3	-109.45	74.99	117.27
Tip	-130.00	60.00	0.00

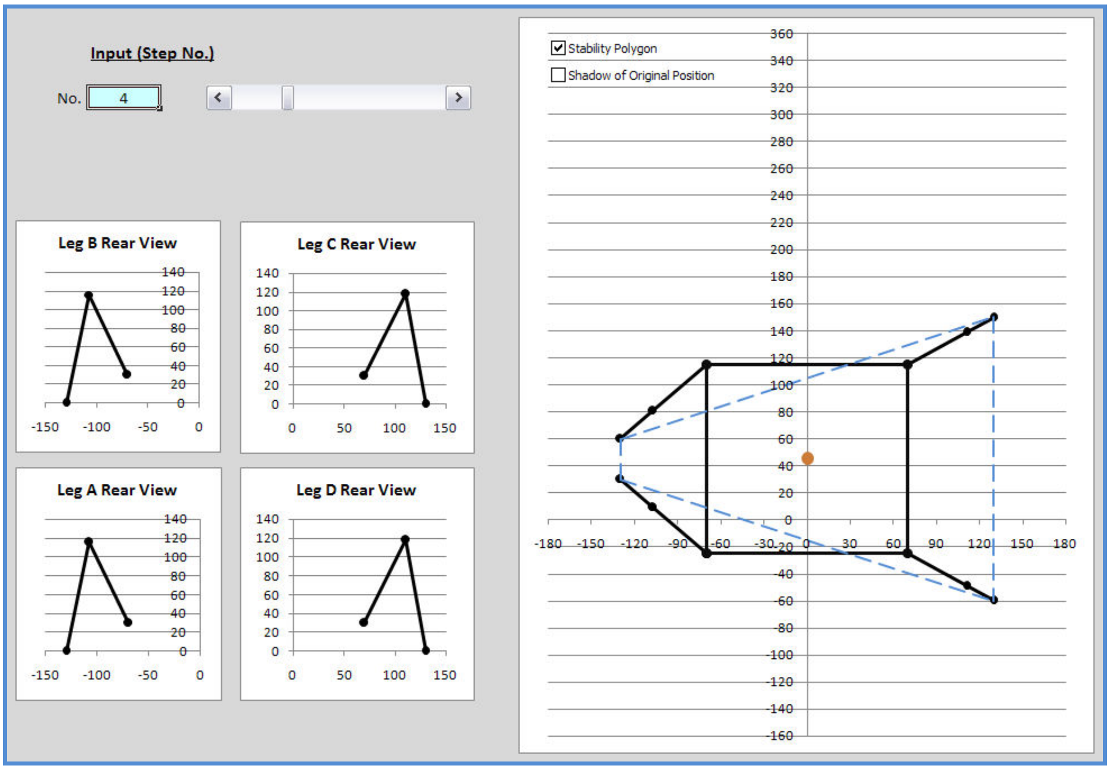
Leg C	x	y	z
Joint 2	70.00	103.75	30.00
Joint 3	109.00	133.81	117.03
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-36.25	30.00
Joint 3	-73.82	-35.91	129.93
Tip	-140.00	-30.00	30.00

Leg D	x	y	z
Joint 2	70.00	-36.25	30.00
Joint 3	113.09	-53.31	118.61
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-5.1	36.1	37.6	-21.6
θ2 °	87.8	60.8	60.5	62.4
θ3 °	144.2	138.5	137.7	143.7

Step 4



Input (Step No.)

No. 4

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	115.00	30.00
Joint 3	-107.53	80.59	116.07
Tip	-130.00	60.00	0.00

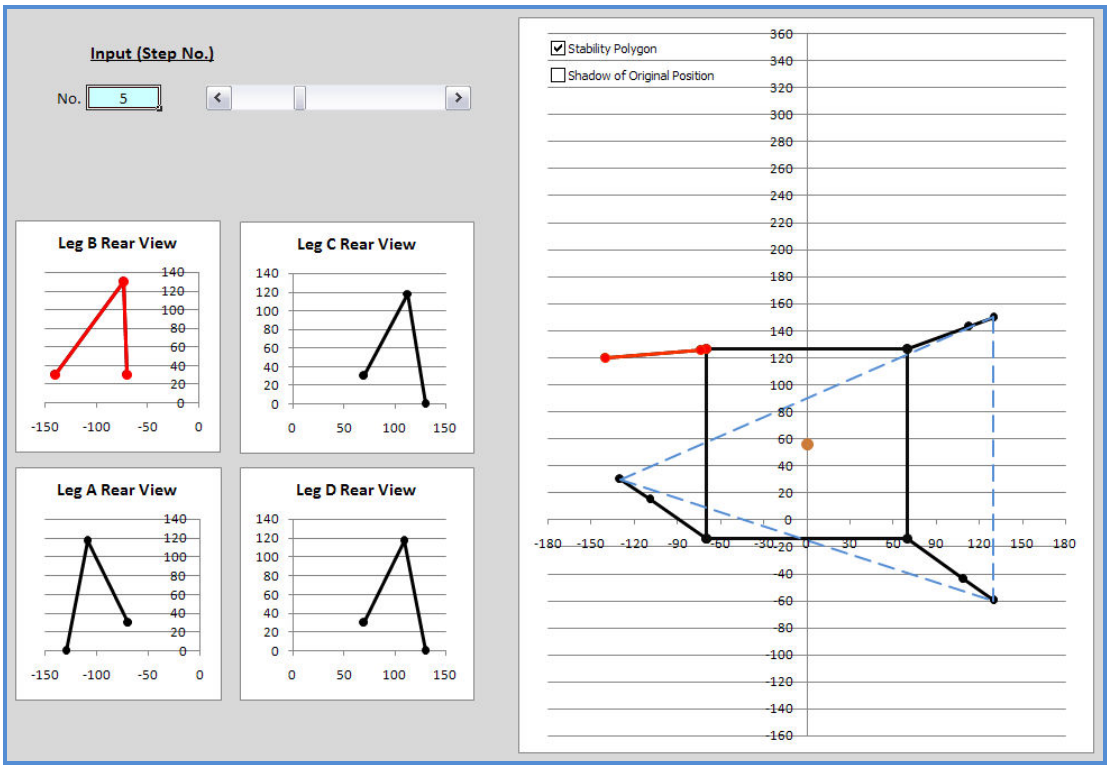
Leg C	x	y	z
Joint 2	70.00	115.00	30.00
Joint 3	111.06	138.95	117.98
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-25.00	30.00
Joint 3	-107.53	9.41	116.07
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	-25.00	30.00
Joint 3	111.06	-48.95	117.98
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-42.5	42.5	30.3	-30.3
θ2 °	59.4	59.4	61.6	61.6
θ3 °	134.7	134.7	141.1	141.1

Step 5



Input (Step No.)

No. 5

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	126.25	30.00
Joint 3	-73.82	125.91	129.93
Tip	-140.00	120.00	30.00

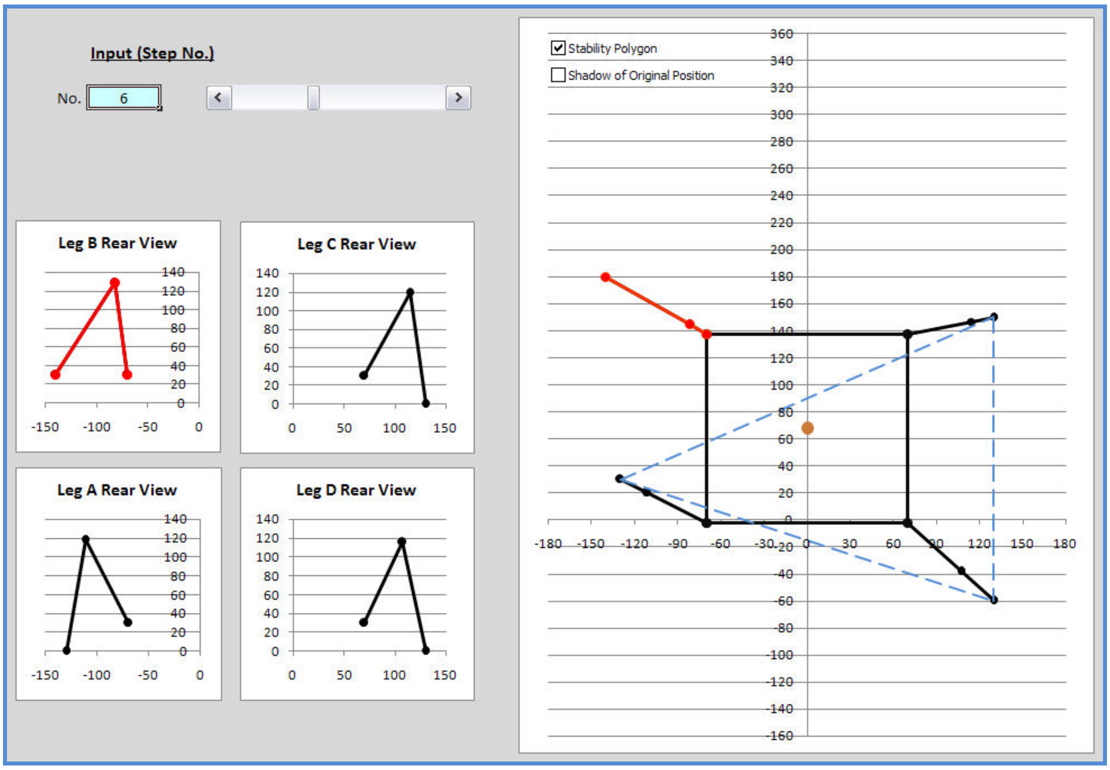
Leg C	x	y	z
Joint 2	70.00	126.25	30.00
Joint 3	113.09	143.31	118.61
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-13.75	30.00
Joint 3	-109.45	15.01	117.27
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	-13.75	30.00
Joint 3	109.00	-43.81	117.03
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-36.1	5.1	21.6	-37.6
θ2 °	60.8	87.8	62.4	60.5
θ3 °	138.5	144.2	143.7	137.7

Step 6



Input (Step No.)

No. 6

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	137.50	30.00
Joint 3	-82.04	144.81	129.00
Tip	-140.00	180.00	30.00

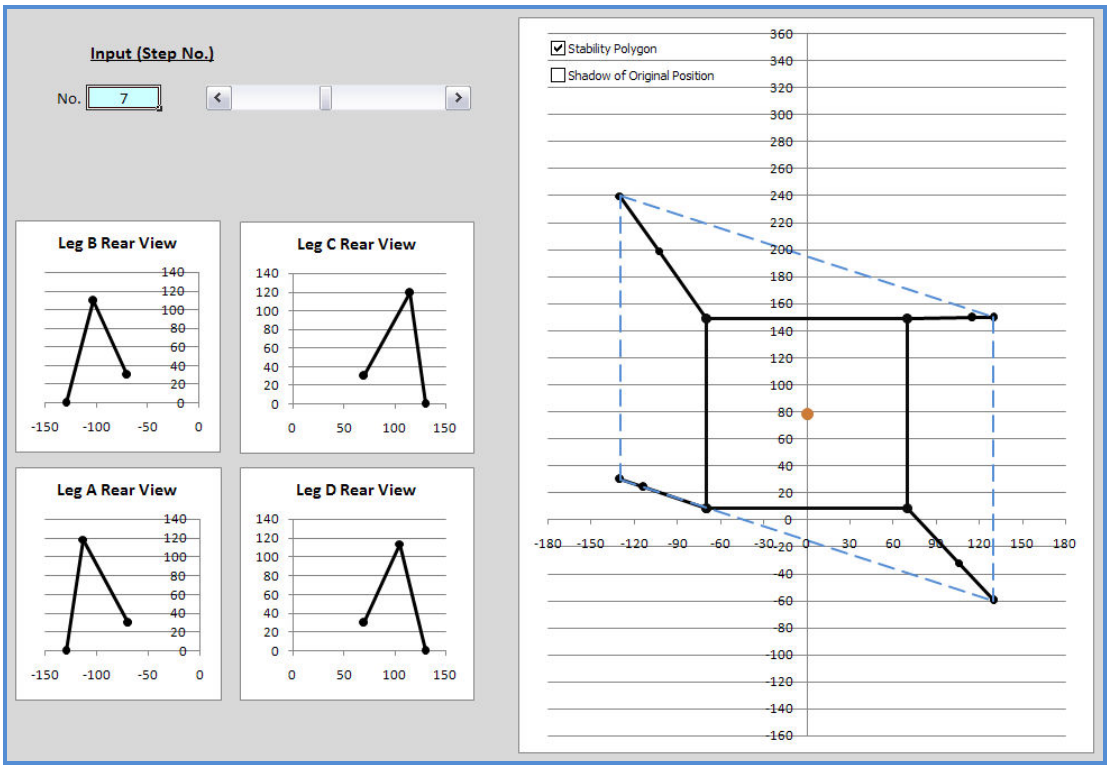
Leg C	x	y	z
Joint 2	70.00	137.50	30.00
Joint 3	114.68	146.81	118.98
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	-2.50	30.00
Joint 3	-111.52	19.99	118.15
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	-2.50	30.00
Joint 3	107.15	-38.10	115.75
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-28.4	-31.3	11.8	-43.8
θ2 °	61.8	81.9	62.8	59.0
θ3 °	141.7	137.5	145.4	133.7

Step 7



Input (Step No.)

No. 7

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	148.75	30.00
Joint 3	-103.22	199.27	109.65
Tip	-130.00	240.00	0.00

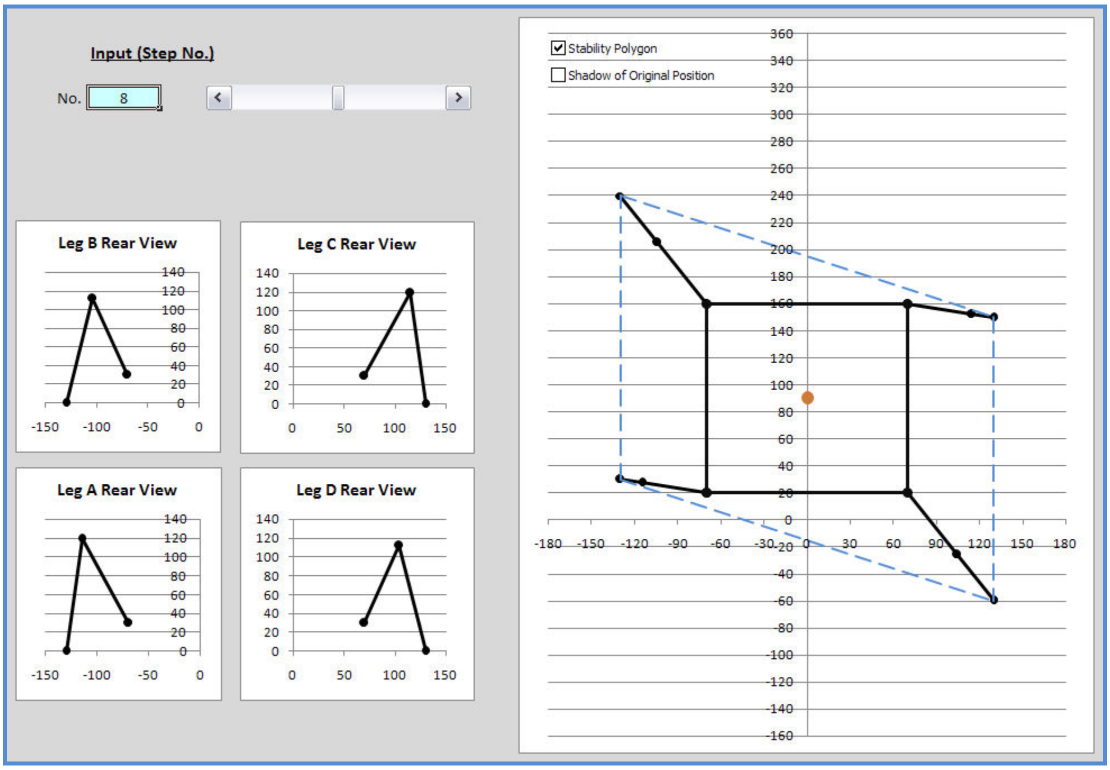
Leg C	x	y	z
Joint 2	70.00	148.75	30.00
Joint 3	115.38	149.70	119.11
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	8.75	30.00
Joint 3	-113.50	24.16	118.72
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	8.75	30.00
Joint 3	105.57	-32.01	114.10
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-19.5	-56.7	1.2	-48.9
θ2 °	62.5	52.8	63.0	57.2
θ3 °	144.1	118.8	146.0	129.2

Step 8



Input (Step No.)

No. 8

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	160.00	30.00
Joint 3	-104.27	205.70	112.08
Tip	-130.00	240.00	0.00

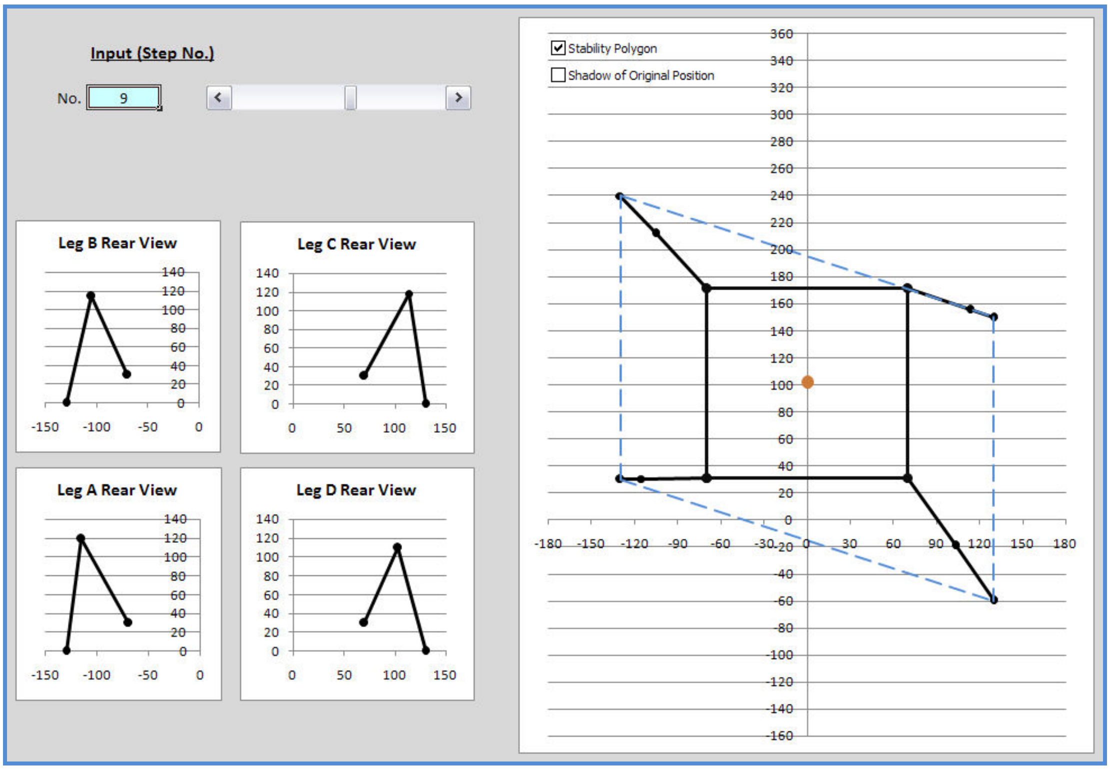
Leg C	x	y	z
Joint 2	70.00	160.00	30.00
Joint 3	114.93	152.51	119.02
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	20.00	30.00
Joint 3	-114.93	27.49	119.02
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	20.00	30.00
Joint 3	104.27	-25.70	112.08
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-9.5	-53.1	-9.5	-53.1
θ2 °	62.9	55.2	62.9	55.2
θ3 °	145.6	124.2	145.6	124.2

Step 9


Input (Step No.)

No. 9

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	171.25	30.00
Joint 3	-105.57	212.01	114.10
Tip	-130.00	240.00	0.00

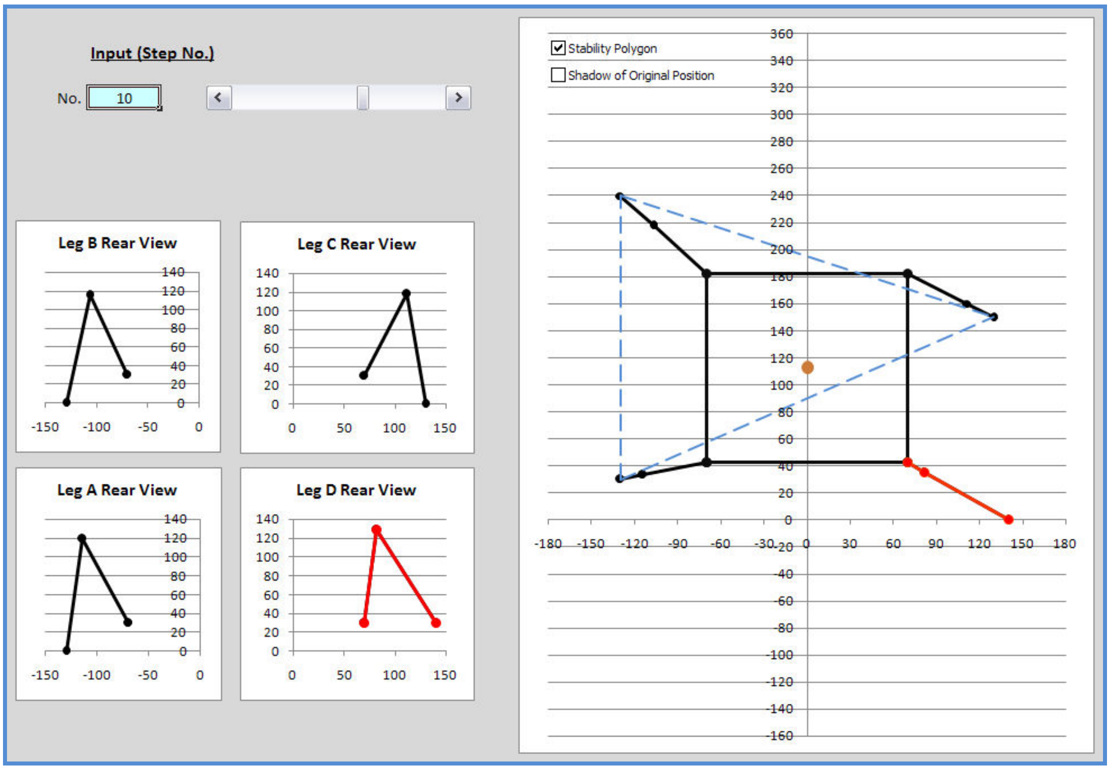
Leg C	x	y	z
Joint 2	70.00	171.25	30.00
Joint 3	113.50	155.84	118.72
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	31.25	30.00
Joint 3	-115.38	30.30	119.11
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	31.25	30.00
Joint 3	103.22	-19.27	109.65
Tip	130.00	-60.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	1.2	-48.9	-19.5	-56.7
θ2 °	63.0	57.2	62.5	52.8
θ3 °	146.0	129.2	144.1	118.8

Step 10



Input (Step No.)

No. 10

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	182.50	30.00
Joint 3	-107.15	218.10	115.75
Tip	-130.00	240.00	0.00

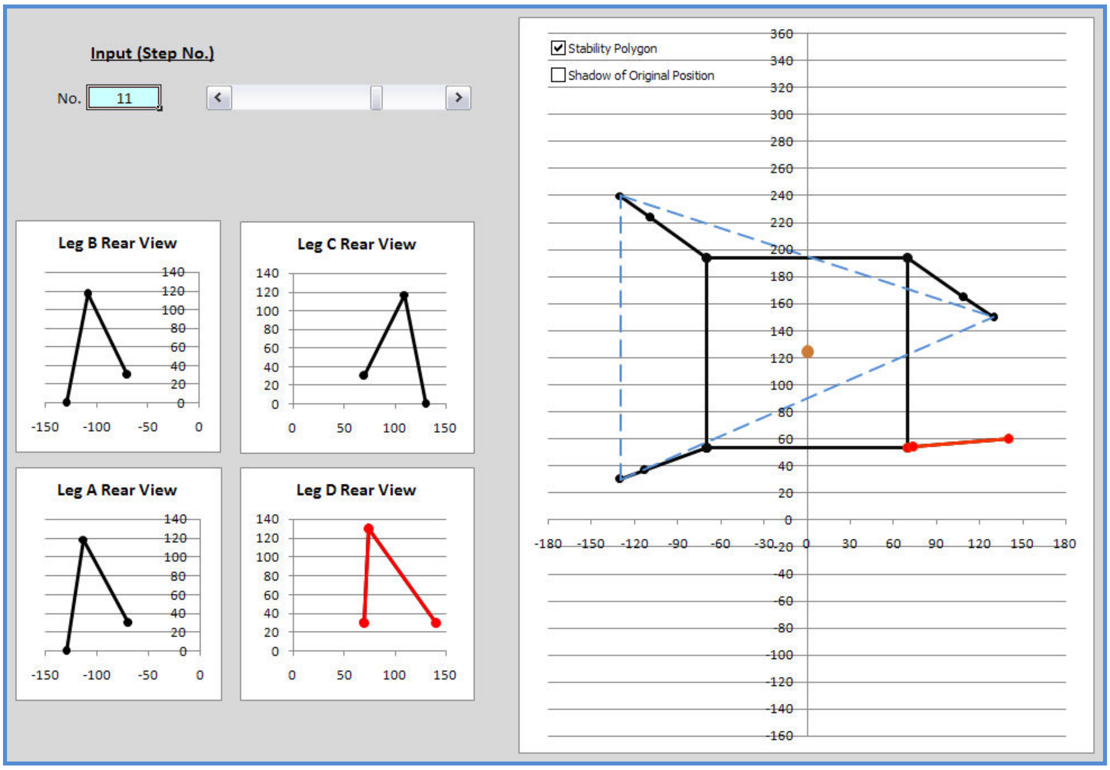
Leg C	x	y	z
Joint 2	70.00	182.50	30.00
Joint 3	111.52	160.01	118.15
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	42.50	30.00
Joint 3	-114.68	33.19	118.98
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	42.50	30.00
Joint 3	82.04	35.19	129.00
Tip	140.00	0.00	30.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	11.8	-43.8	-28.4	-31.3
θ2 °	62.8	59.0	61.8	81.9
θ3 °	145.4	133.7	141.7	137.5

Step 11



Leg B	x	y	z
Joint 2	-70.00	193.75	30.00
Joint 3	-109.00	223.81	117.03
Tip	-130.00	240.00	0.00

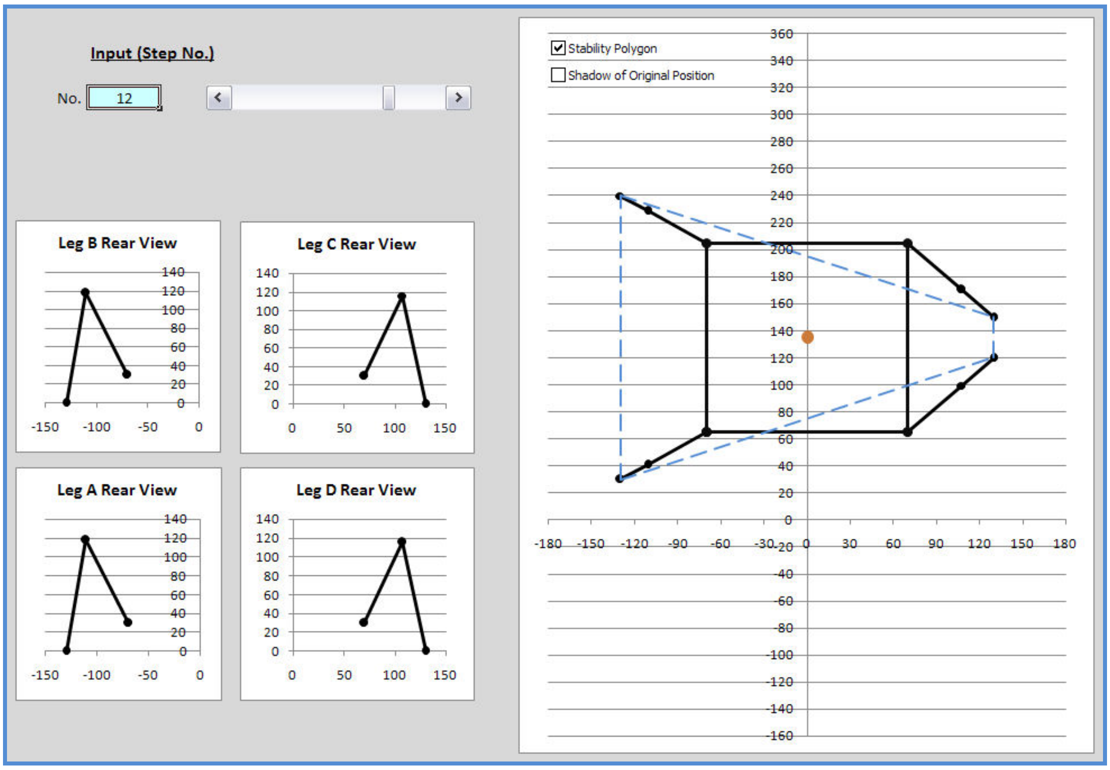
Leg C	x	y	z
Joint 2	70.00	193.75	30.00
Joint 3	109.45	164.99	117.27
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	53.75	30.00
Joint 3	-113.09	36.69	118.61
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	53.75	30.00
Joint 3	73.82	54.09	129.93
Tip	140.00	60.00	30.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	21.6	-37.6	-36.1	5.1
θ2 °	62.4	60.5	60.8	87.8
θ3 °	143.7	137.7	138.5	144.2

Step 12



Leg B	x	y	z
Joint 2	-70.00	205.00	30.00
Joint 3	-111.06	228.95	117.98
Tip	-130.00	240.00	0.00

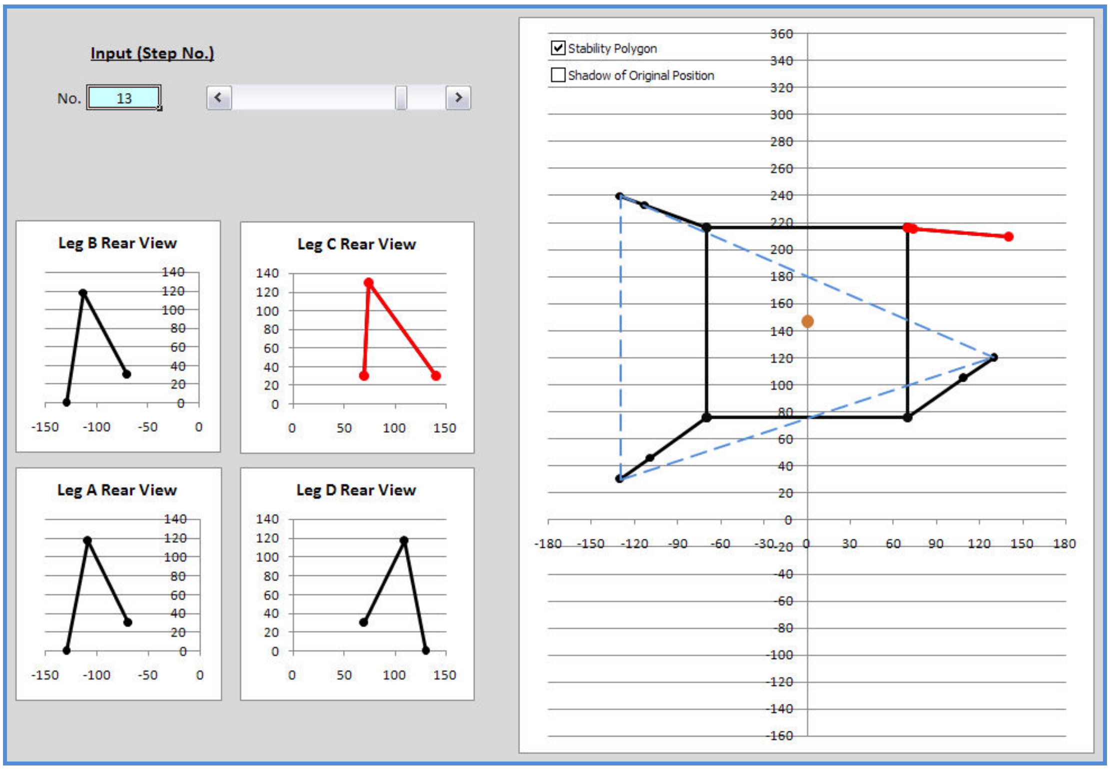
Leg C	x	y	z
Joint 2	70.00	205.00	30.00
Joint 3	107.53	170.59	116.07
Tip	130.00	150.00	0.00

Leg A	x	y	z
Joint 2	-70.00	65.00	30.00
Joint 3	-111.06	41.05	117.98
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	65.00	30.00
Joint 3	107.53	99.41	116.07
Tip	130.00	120.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	30.3	-30.3	-42.5	42.5
θ2 °	61.6	61.6	59.4	59.4
θ3 °	141.1	141.1	134.7	134.7

Step 13



Input (Step No.)

No. 13

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	216.25	30.00
Joint 3	-113.09	233.31	118.61
Tip	-130.00	240.00	0.00

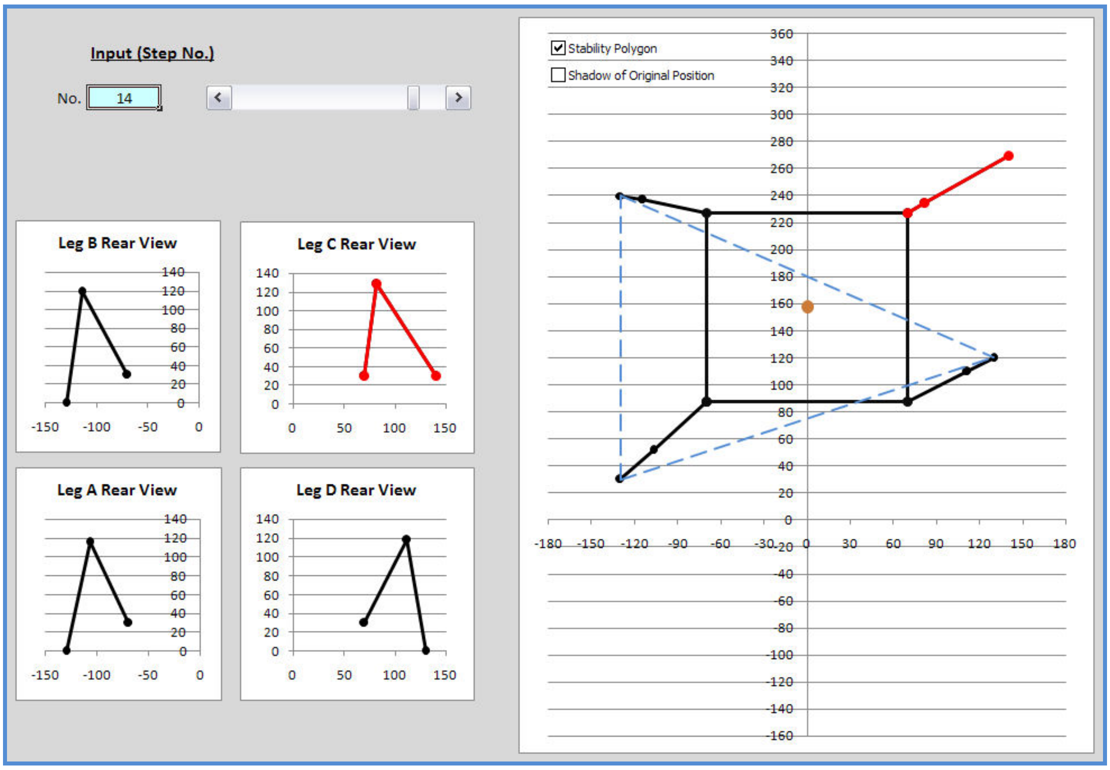
Leg C	x	y	z
Joint 2	70.00	216.25	30.00
Joint 3	73.82	215.91	129.93
Tip	140.00	210.00	30.00

Leg A	x	y	z
Joint 2	-70.00	76.25	30.00
Joint 3	-109.00	46.19	117.03
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	76.25	30.00
Joint 3	109.45	105.01	117.27
Tip	130.00	120.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	37.6	-21.6	-5.1	36.1
θ2 °	60.5	62.4	87.8	60.8
θ3 °	137.7	143.7	144.2	138.5

Step 14



Leg B	x	y	z
Joint 2	-70.00	227.50	30.00
Joint 3	-114.68	236.81	118.98
Tip	-130.00	240.00	0.00

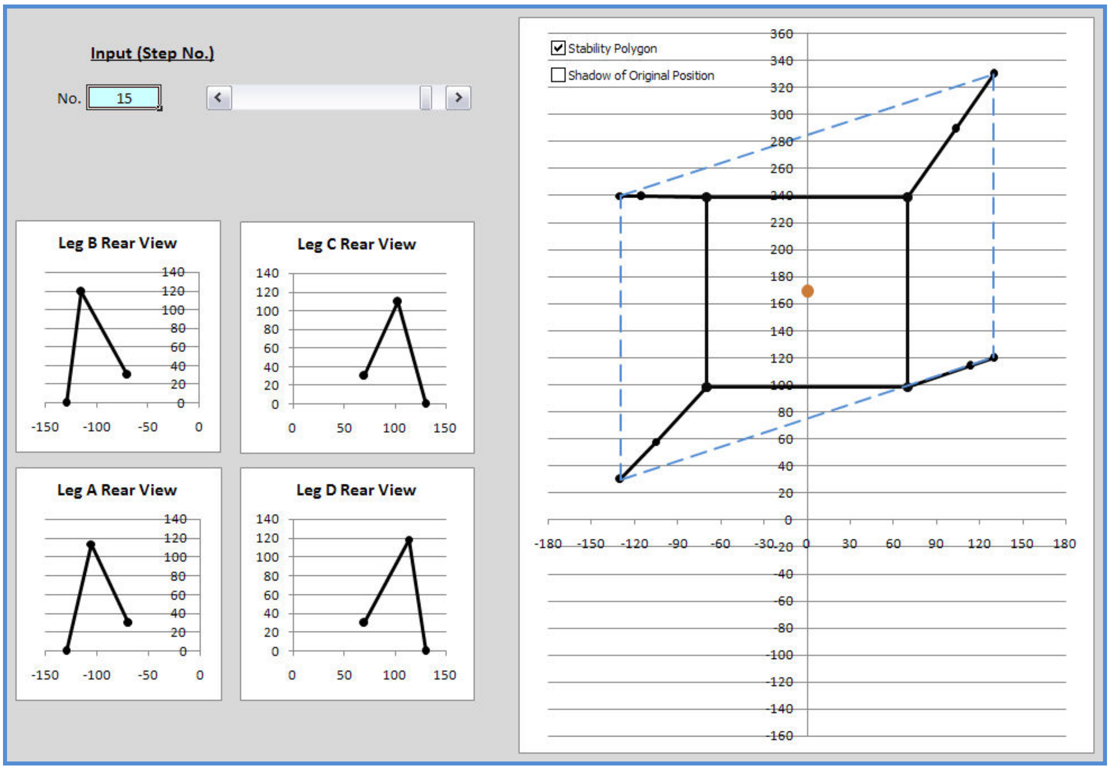
Leg C	x	y	z
Joint 2	70.00	227.50	30.00
Joint 3	82.04	234.81	129.00
Tip	140.00	270.00	30.00

Leg A	x	y	z
Joint 2	-70.00	87.50	30.00
Joint 3	-107.15	51.90	115.75
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	87.50	30.00
Joint 3	111.52	109.99	118.15
Tip	130.00	120.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	43.8	-11.8	31.3	28.4
θ2 °	59.0	62.8	81.9	61.8
θ3 °	133.7	145.4	137.5	141.7

Step 15



Input (Step No.)

No. 15

☑ Stability Polygon

☐ Shadow of Original Position

Leg B	x	y	z
Joint 2	-70.00	238.75	30.00
Joint 3	-115.38	239.70	119.11
Tip	-130.00	240.00	0.00

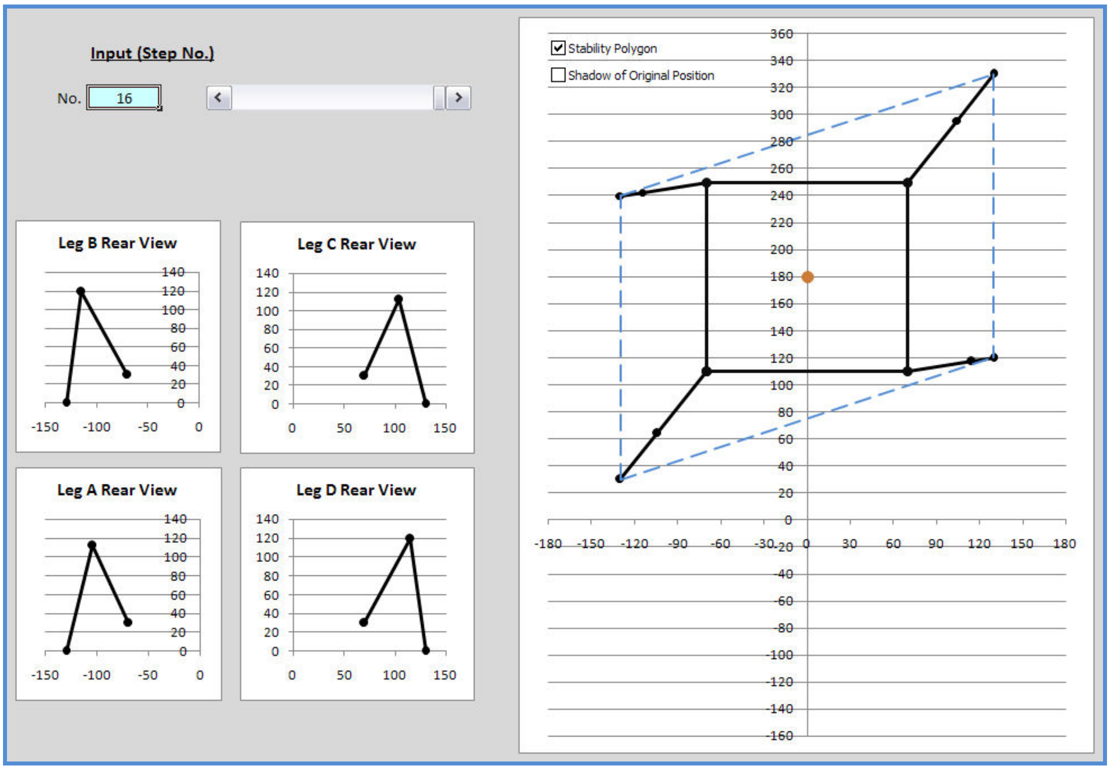
Leg C	x	y	z
Joint 2	70.00	238.75	30.00
Joint 3	103.22	289.27	109.65
Tip	130.00	330.00	0.00

Leg A	x	y	z
Joint 2	-70.00	98.75	30.00
Joint 3	-105.57	57.99	114.10
Tip	-130.00	30.00	0.00

Leg D	x	y	z
Joint 2	70.00	98.75	30.00
Joint 3	113.50	114.16	118.72
Tip	130.00	120.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	48.9	-1.2	56.7	19.5
θ2 °	57.2	63.0	52.8	62.5
θ3 °	129.2	146.0	118.8	144.1

Step 16



Leg B	x	y	z
Joint 2	-70.00	250.00	30.00
Joint 3	-114.93	242.51	119.02
Tip	-130.00	240.00	0.00

Leg C	x	y	z
Joint 2	70.00	250.00	30.00
Joint 3	104.27	295.70	112.08
Tip	130.00	330.00	0.00

Leg A	x	y	z
Joint 2	-70.00	110.00	30.00
Joint 3	-104.27	64.30	112.08
Tip	-130.00	30.00	0.00

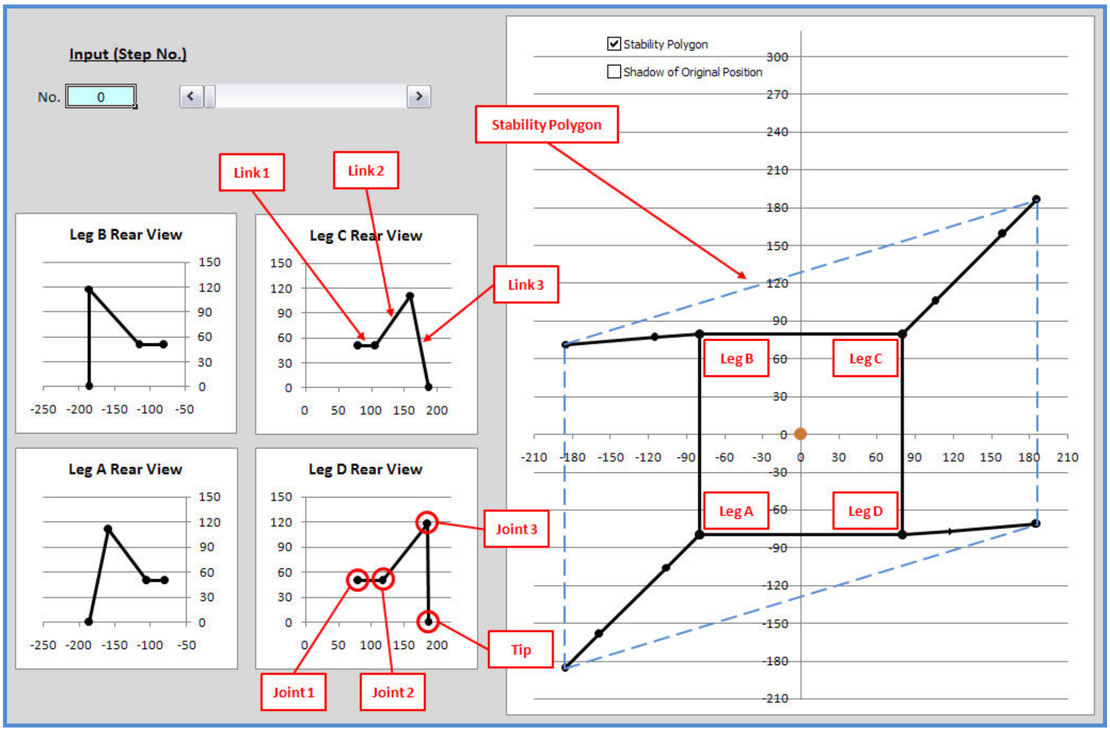
Leg D	x	y	z
Joint 2	70.00	110.00	30.00
Joint 3	114.93	117.49	119.02
Tip	130.00	120.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	53.1	9.5	53.1	9.5
θ2 °	55.2	62.9	55.2	62.9
θ3 °	124.2	145.6	124.2	145.6

APPENDIX 15

Readjusted Gait Simulation

Step 0 (Initial Position)



Input (Step No.)

No. 0

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	80.00	50.00
Joint 2	-115.60	77.05	50.00
Joint 3	-185.43	71.40	118.00
Tip	-186.07	71.35	0.00

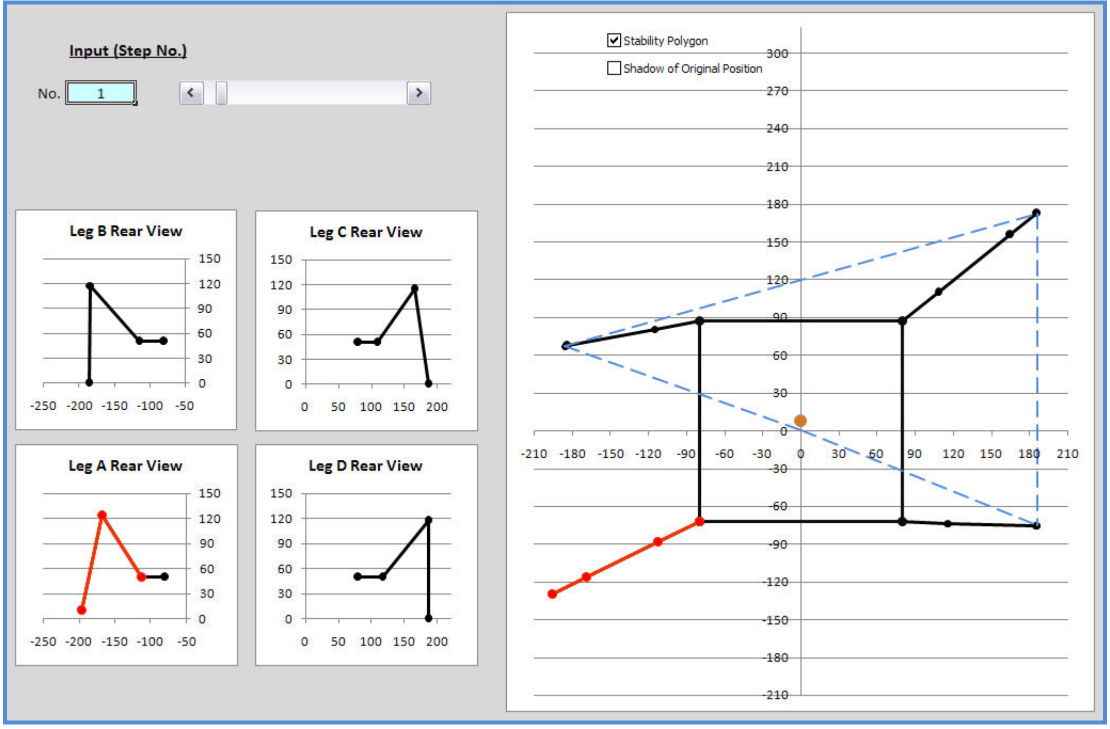
Leg C	x	y	z
Joint 1	80.00	80.00	50.00
Joint 2	105.63	105.63	50.00
Joint 3	158.87	158.87	111.56
Tip	186.07	186.07	0.00

Leg A	x	y	z
Joint 1	-80.00	-80.00	50.00
Joint 2	-105.63	-105.63	50.00
Joint 3	-158.87	-158.87	111.56
Tip	-186.07	-186.07	0.00

Leg D	x	y	z
Joint 1	80.00	-80.00	50.00
Joint 2	116.13	-77.05	50.00
Joint 3	185.43	-71.40	118.00
Tip	186.07	-71.35	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	45.0	4.7	45.0	4.7
θ2 °	39.3	44.4	39.3	44.4
θ3 °	110.2	134.1	110.2	134.1

Step 1



Input (Step No.)

No. 1

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	87.50	50.00
Joint 2	-115.60	80.66	50.00
Joint 3	-183.90	67.55	117.98
Tip	-186.07	67.13	0.00

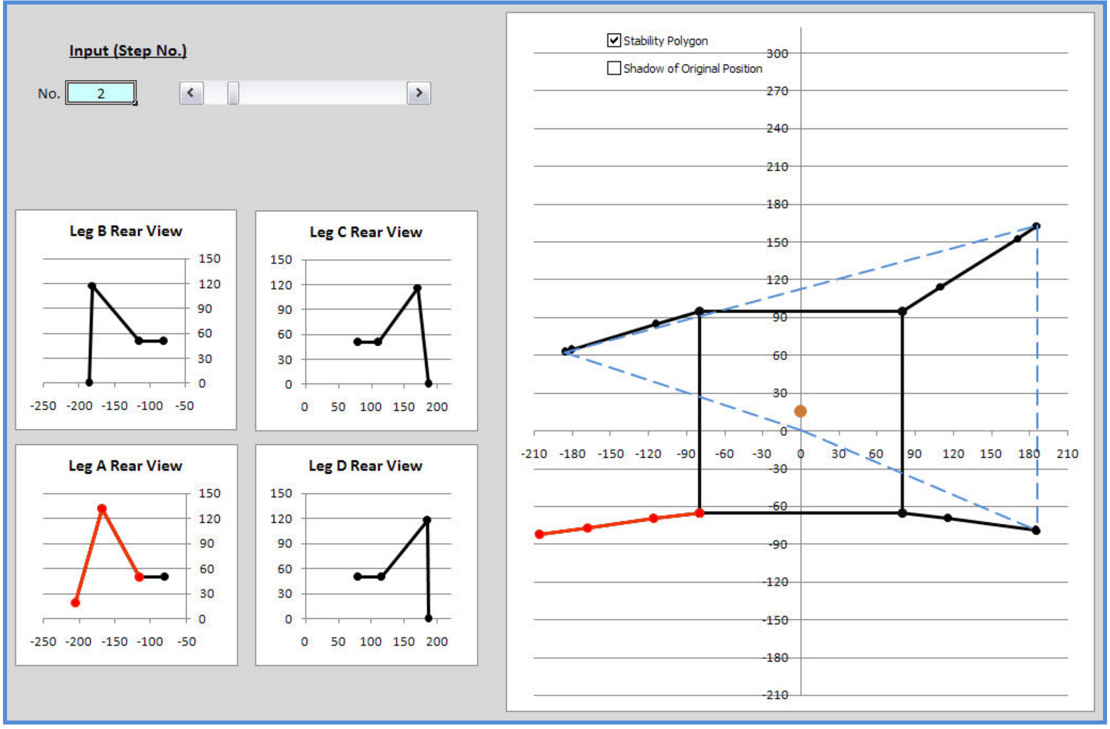
Leg C	x	y	z
Joint 1	80.00	87.50	50.00
Joint 2	108.25	110.21	50.00
Joint 3	164.78	155.64	114.80
Tip	186.07	172.75	0.00

Leg A	x	y	z
Joint 1	-80.00	-72.50	50.00
Joint 2	-112.48	-88.60	50.00
Joint 3	-168.95	-116.61	124.05
Tip	-196.07	-130.06	10.00

Leg D	x	y	z
Joint 1	80.00	-72.50	50.00
Joint 2	116.24	-73.48	50.00
Joint 3	185.74	-75.35	118.00
Tip	186.07	-75.36	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	26.4	10.9	38.8	-1.5
θ2 °	49.6	44.3	41.8	44.4
θ3 °	124.7	133.3	118.4	134.2

Step 2



Input (Step No.)

No. 2

☑ Stability Polygon

☐ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	95.00	50.00
Joint 2	-114.65	84.35	50.00
Joint 3	-181.21	63.90	117.89
Tip	-186.07	62.41	0.00

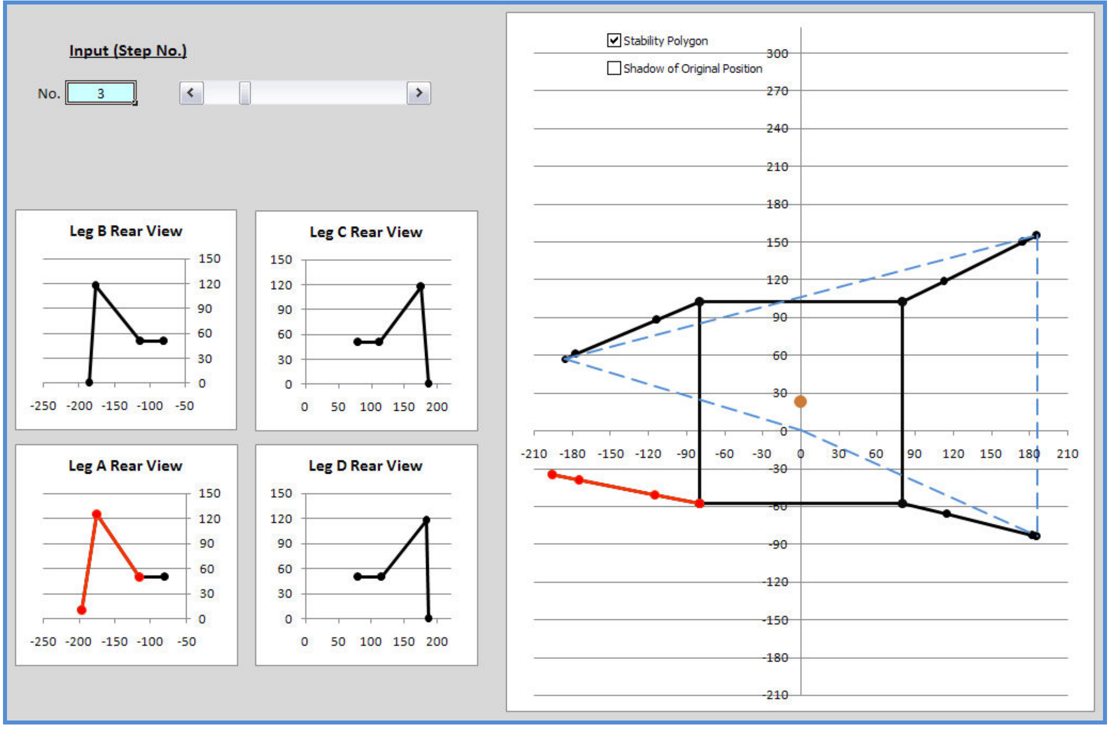
Leg C	x	y	z
Joint 1	80.00	95.00	50.00
Joint 2	110.54	114.52	50.00
Joint 3	170.33	152.73	116.51
Tip	186.07	162.79	0.00

Leg A	x	y	z
Joint 1	-80.00	-65.00	50.00
Joint 2	-115.92	-69.89	50.00
Joint 3	-168.25	-77.01	131.66
Tip	-206.07	-82.16	20.00

Leg D	x	y	z
Joint 1	80.00	-65.00	50.00
Joint 2	115.92	-69.89	50.00
Joint 3	184.81	-79.27	117.99
Tip	186.07	-79.44	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	7.8	17.1	32.6	-7.8
θ2 °	57.1	44.3	43.2	44.4
θ3 °	128.2	131.8	124.0	133.7

Step 3



Input (Step No.)

No. 3

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	102.50	50.00
Joint 2	-113.30	88.17	50.00
Joint 3	-177.49	60.54	117.63
Tip	-186.07	56.85	0.00

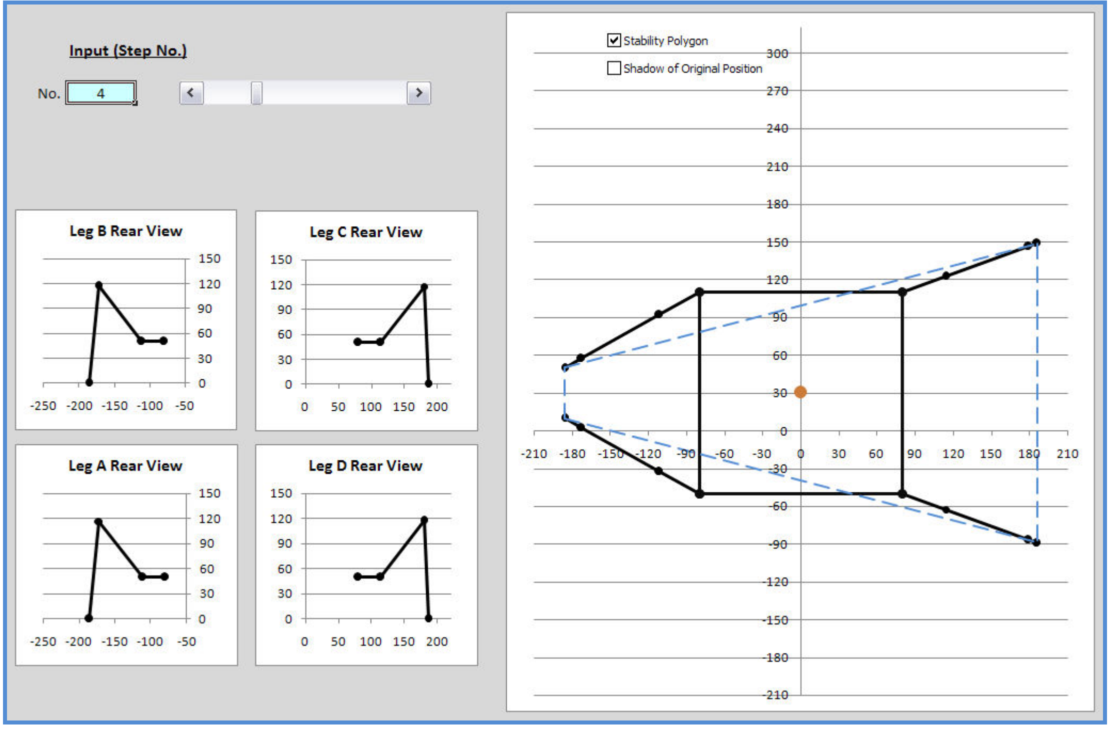
Leg C	x	y	z
Joint 1	80.00	102.50	50.00
Joint 2	112.48	118.60	50.00
Joint 3	175.30	149.76	117.39
Tip	186.07	155.10	0.00

Leg A	x	y	z
Joint 1	-80.00	-57.50	50.00
Joint 2	-115.60	-50.66	50.00
Joint 3	-175.12	-39.23	126.06
Tip	-196.07	-35.21	10.00

Leg D	x	y	z
Joint 1	80.00	-57.50	50.00
Joint 2	115.18	-66.25	50.00
Joint 3	182.70	-83.03	117.95
Tip	186.07	-83.87	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-10.9	23.3	26.4	-14.0
θ2 °	51.5	44.1	43.9	44.3
θ3 °	131.0	129.5	128.0	132.6

Step 4



Input (Step No.)

No. 4

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	110.00	50.00
Joint 2	-111.55	92.15	50.00
Joint 3	-172.88	57.46	117.02
Tip	-186.07	50.00	0.00

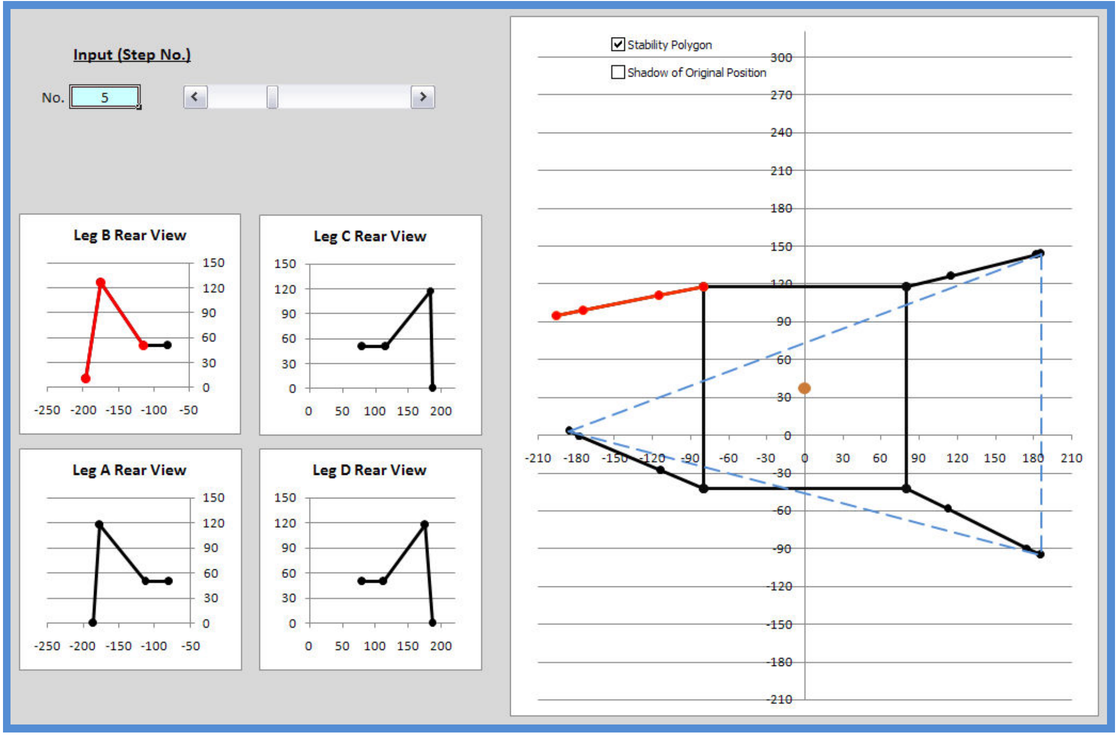
Leg C	x	y	z
Joint 1	80.00	110.00	50.00
Joint 2	114.03	122.50	50.00
Joint 3	179.48	146.54	117.79
Tip	186.07	148.96	0.00

Leg A	x	y	z
Joint 1	-80.00	-50.00	50.00
Joint 2	-111.55	-32.15	50.00
Joint 3	-172.88	2.54	117.02
Tip	-186.07	10.00	0.00

Leg D	x	y	z
Joint 1	80.00	-50.00	50.00
Joint 2	114.03	-62.50	50.00
Joint 3	179.48	-86.54	117.79
Tip	186.07	-88.96	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-29.5	29.5	20.2	-20.2
θ2 °	43.6	43.6	44.2	44.2
θ3 °	126.2	126.2	130.8	130.8

Step 5



Input (Step No.)

No. 5

☑ Stability Polygon

□ Shadow of Original Position

Leg B Rear View

Leg C Rear View

Leg A Rear View

Leg D Rear View

Leg B	x	y	z
Joint 1	-80.00	117.50	50.00
Joint 2	-115.60	110.66	50.00
Joint 3	-175.12	99.23	126.06
Tip	-196.07	95.21	10.00

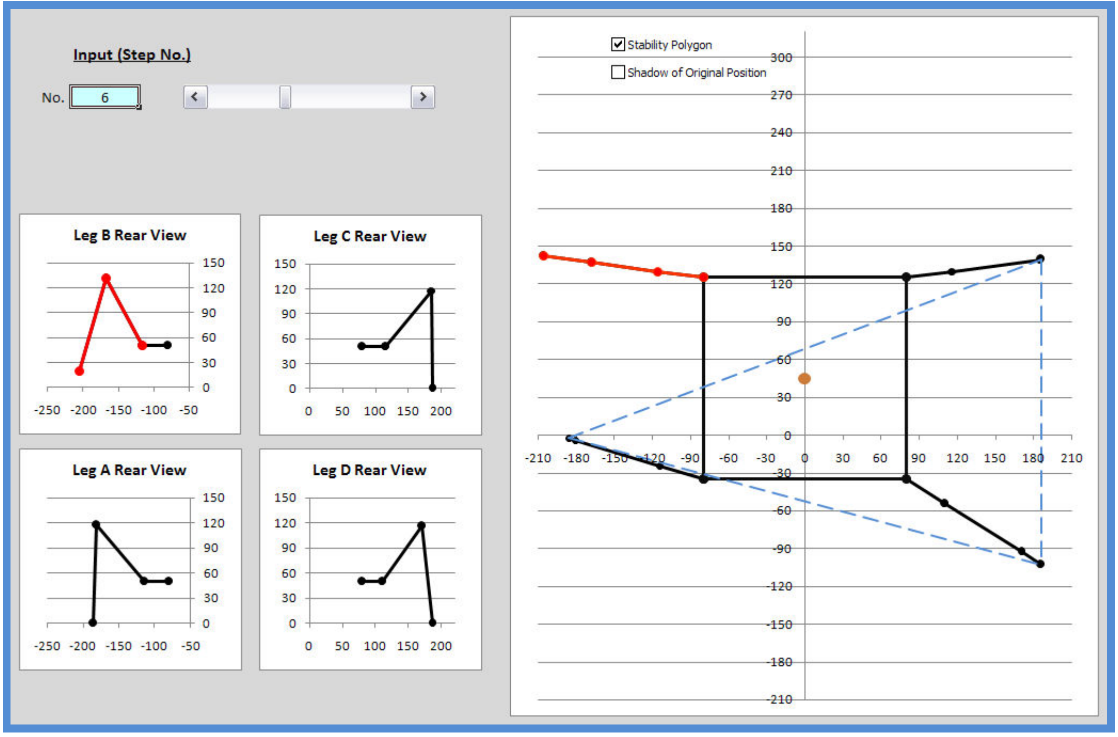
Leg C	x	y	z
Joint 1	80.00	117.50	50.00
Joint 2	115.18	126.25	50.00
Joint 3	182.70	143.03	117.95
Tip	186.07	143.87	0.00

Leg A	x	y	z
Joint 1	-80.00	-42.50	50.00
Joint 2	-113.30	-28.17	50.00
Joint 3	-177.49	-0.54	117.63
Tip	-186.07	3.15	0.00

Leg D	x	y	z
Joint 1	80.00	-42.50	50.00
Joint 2	112.48	-58.60	50.00
Joint 3	175.30	-89.76	117.39
Tip	186.07	-95.10	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-23.3	10.9	14.0	-26.4
θ2 °	44.1	51.5	44.3	43.9
θ3 °	129.5	131.0	132.6	128.0

Step 6



Input (Step No.)

No. 6

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	125.00	50.00
Joint 2	-115.92	129.89	50.00
Joint 3	-168.25	137.01	131.66
Tip	-206.07	142.16	20.00

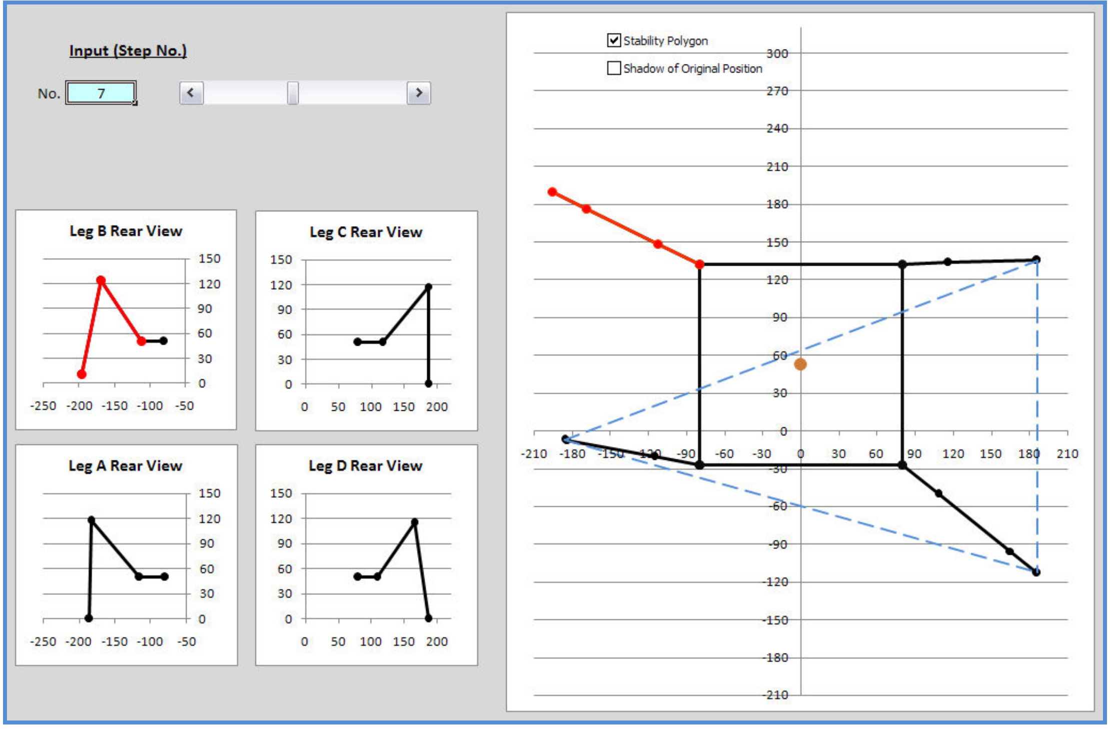
Leg C	x	y	z
Joint 1	80.00	125.00	50.00
Joint 2	115.92	129.89	50.00
Joint 3	184.81	139.27	117.99
Tip	186.07	139.44	0.00

Leg A	x	y	z
Joint 1	-80.00	-35.00	50.00
Joint 2	-114.65	-24.35	50.00
Joint 3	-181.21	-3.90	117.89
Tip	-186.07	-2.41	0.00

Leg D	x	y	z
Joint 1	80.00	-35.00	50.00
Joint 2	110.54	-54.52	50.00
Joint 3	170.33	-92.73	116.51
Tip	186.07	-102.79	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-17.1	-7.8	7.8	-32.6
θ2 °	44.3	57.1	44.4	43.2
θ3 °	131.8	128.2	133.7	124.0

Step 7


Input (Step No.)

No. 7

☑ Stability Polygon

☐ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	132.50	50.00
Joint 2	-112.48	148.60	50.00
Joint 3	-168.95	176.61	124.05
Tip	-196.07	190.06	10.00

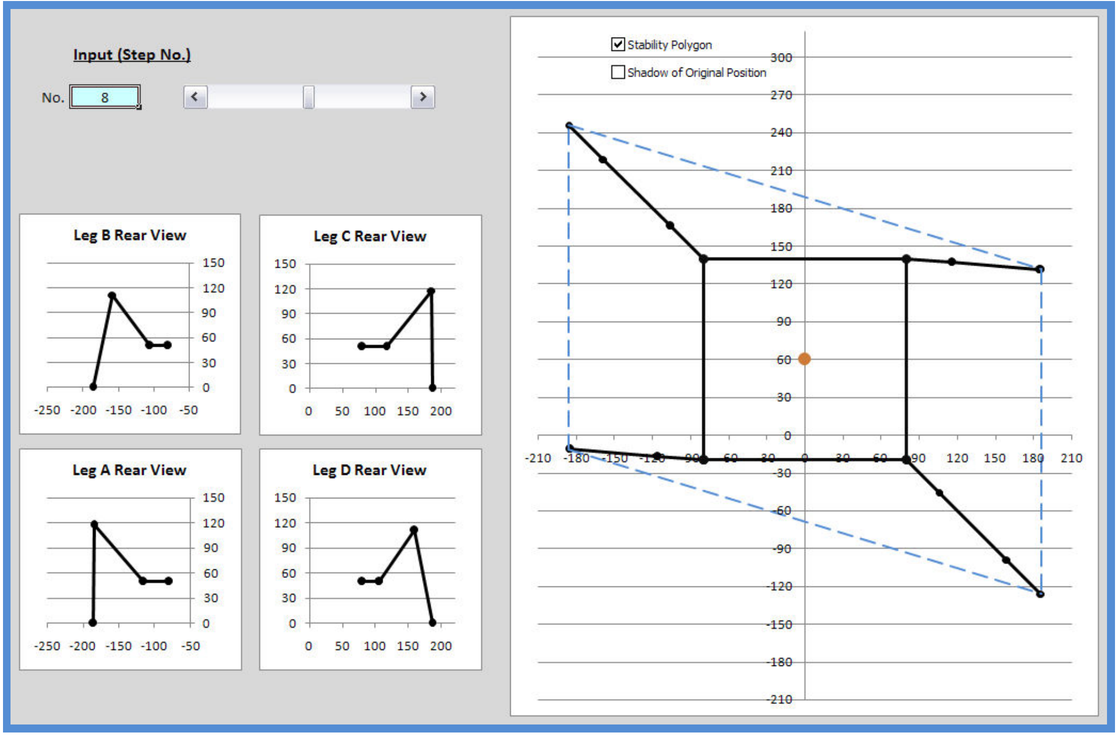
Leg C	x	y	z
Joint 1	80.00	132.50	50.00
Joint 2	116.24	133.48	50.00
Joint 3	185.74	135.35	118.00
Tip	186.07	135.36	0.00

Leg A	x	y	z
Joint 1	-80.00	-27.50	50.00
Joint 2	-115.60	-20.66	50.00
Joint 3	-183.90	-7.55	117.98
Tip	-186.07	-7.13	0.00

Leg D	x	y	z
Joint 1	80.00	-27.50	50.00
Joint 2	108.25	-50.21	50.00
Joint 3	164.78	-95.64	114.80
Tip	186.07	-112.75	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-10.9	-26.4	1.5	-38.8
θ2 °	44.3	49.6	44.4	41.8
θ3 °	133.3	124.7	134.2	118.4

Step 8



Input (Step No.)

No. 8

☑ Stability Polygon

☐ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	140.00	50.00
Joint 2	-105.63	165.63	50.00
Joint 3	-158.87	218.87	111.56
Tip	-186.07	246.07	0.00

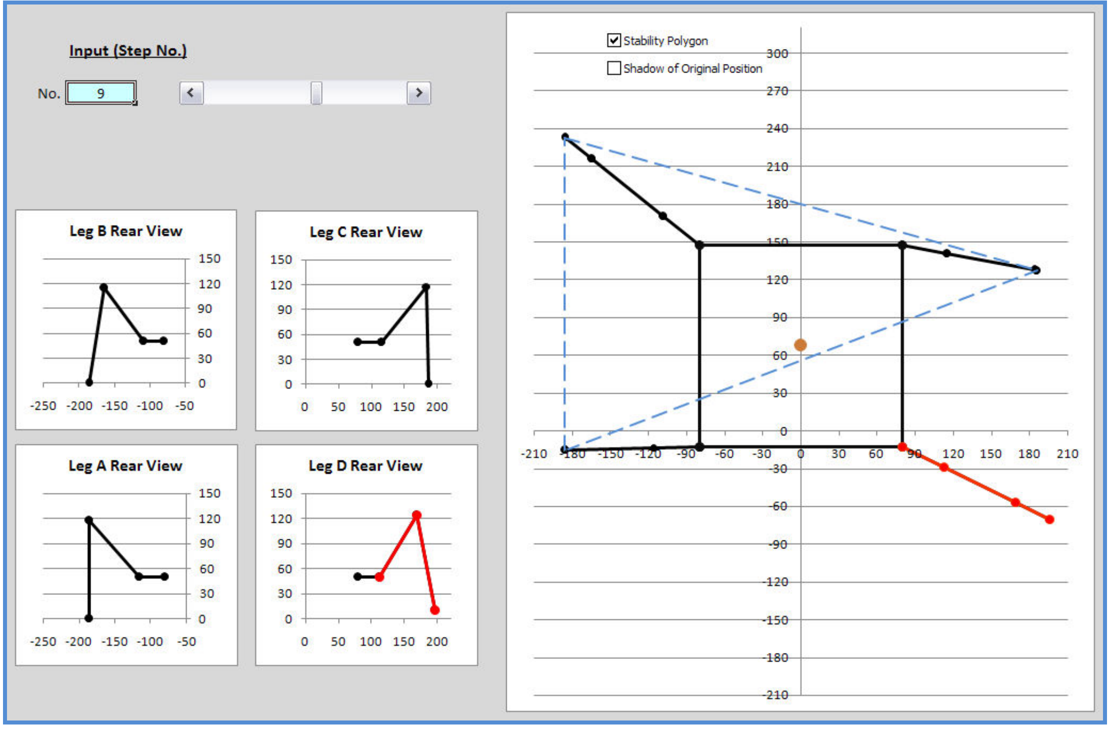
Leg C	x	y	z
Joint 1	80.00	140.00	50.00
Joint 2	116.13	137.05	50.00
Joint 3	185.43	131.40	118.00
Tip	186.07	131.35	0.00

Leg A	x	y	z
Joint 1	-80.00	-20.00	50.00
Joint 2	-116.13	-17.05	50.00
Joint 3	-185.43	-11.40	118.00
Tip	-186.07	-11.35	0.00

Leg D	x	y	z
Joint 1	80.00	-20.00	50.00
Joint 2	105.63	-45.63	50.00
Joint 3	158.87	-98.87	111.56
Tip	186.07	-126.07	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	-4.7	-45.0	-4.7	-45.0
θ2 °	44.4	39.3	44.4	39.3
θ3 °	134.1	110.2	134.1	110.2

Step 9



Input (Step No.)

No. 9

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	147.50	50.00
Joint 2	-108.25	170.21	50.00
Joint 3	-164.78	215.64	114.80
Tip	-186.07	232.75	0.00

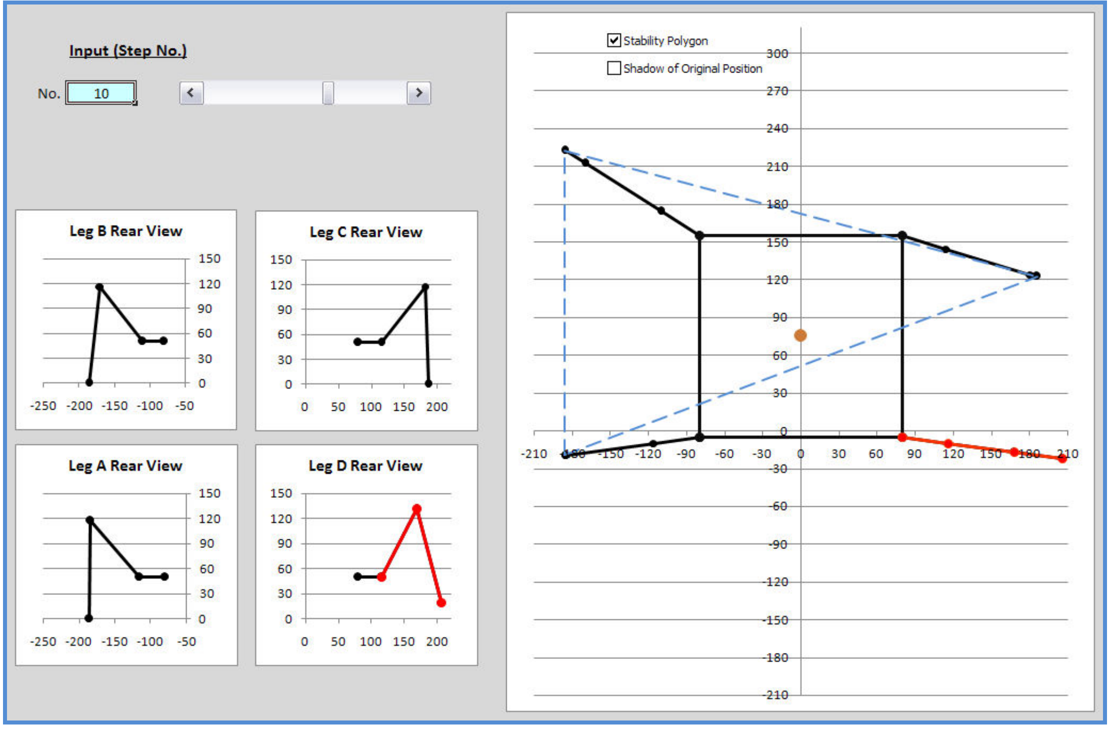
Leg C	x	y	z
Joint 1	80.00	147.50	50.00
Joint 2	115.60	140.66	50.00
Joint 3	183.90	127.55	117.98
Tip	186.07	127.13	0.00

Leg A	x	y	z
Joint 1	-80.00	-12.50	50.00
Joint 2	-116.24	-13.48	50.00
Joint 3	-185.74	-15.35	118.00
Tip	-186.07	-15.36	0.00

Leg D	x	y	z
Joint 1	80.00	-12.50	50.00
Joint 2	112.48	-28.60	50.00
Joint 3	168.95	-56.61	124.05
Tip	196.07	-70.06	10.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	1.5	-38.8	-10.9	-26.4
θ2 °	44.4	41.8	44.3	49.6
θ3 °	134.2	118.4	133.3	124.7

Step 10


Input (Step No.)

No. 10

☑ Stability Polygon

☐ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	155.00	50.00
Joint 2	-110.54	174.52	50.00
Joint 3	-170.33	212.73	116.51
Tip	-186.07	222.79	0.00

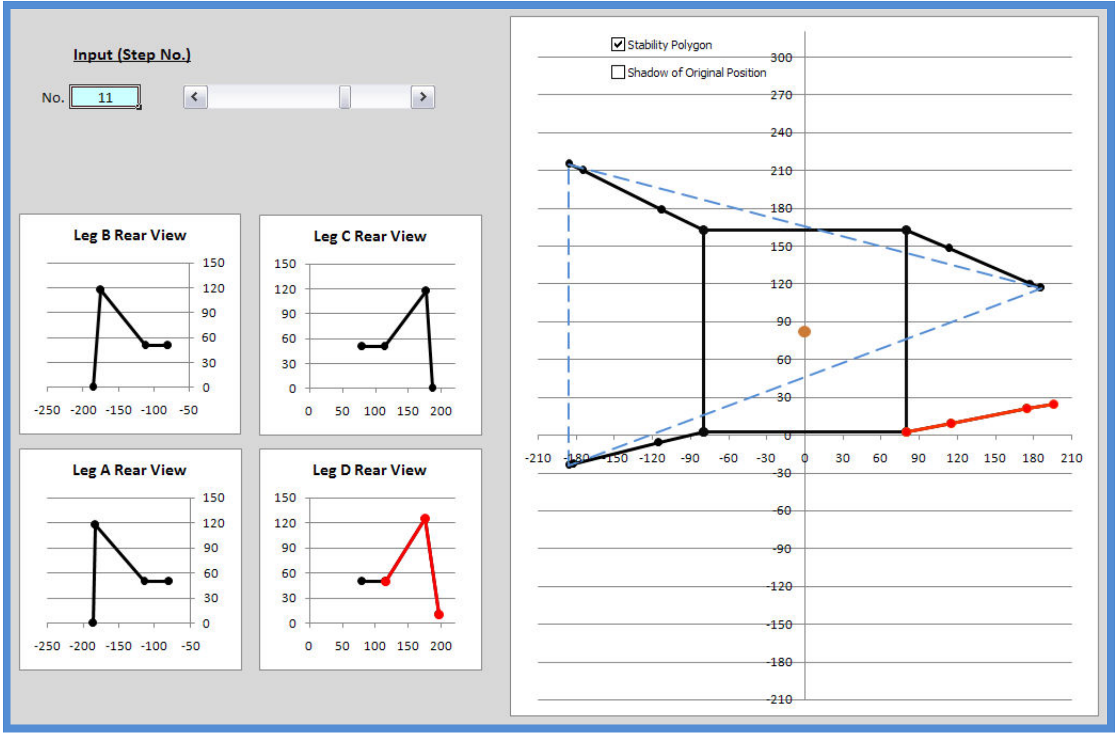
Leg C	x	y	z
Joint 1	80.00	155.00	50.00
Joint 2	114.65	144.35	50.00
Joint 3	181.21	123.90	117.89
Tip	186.07	122.41	0.00

Leg A	x	y	z
Joint 1	-80.00	-5.00	50.00
Joint 2	-115.92	-9.89	50.00
Joint 3	-184.81	-19.27	117.99
Tip	-186.07	-19.44	0.00

Leg D	x	y	z
Joint 1	80.00	-5.00	50.00
Joint 2	115.92	-9.89	50.00
Joint 3	168.25	-17.01	131.66
Tip	206.07	-22.16	20.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	7.8	-32.6	-17.1	-7.8
θ2 °	44.4	43.2	44.3	57.1
θ3 °	133.7	124.0	131.8	128.2

Step 11



Input (Step No.)

No. 11

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	162.50	50.00
Joint 2	-112.48	178.60	50.00
Joint 3	-175.30	209.76	117.39
Tip	-186.07	215.10	0.00

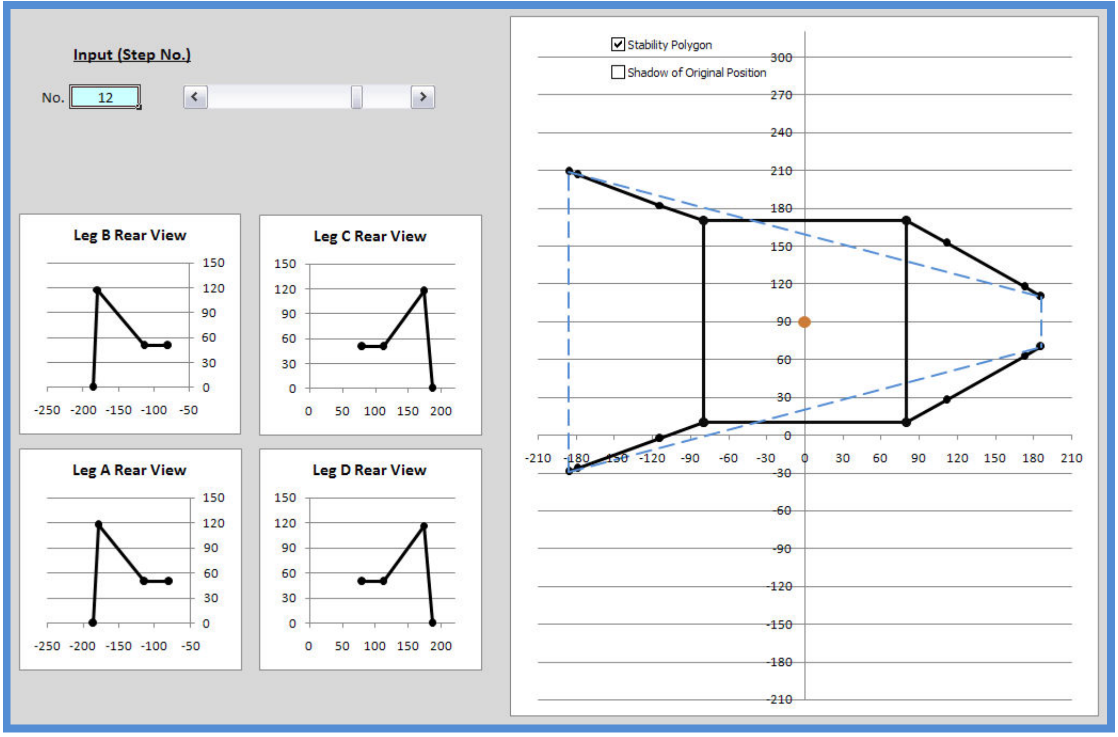
Leg C	x	y	z
Joint 1	80.00	162.50	50.00
Joint 2	113.30	148.17	50.00
Joint 3	177.49	120.54	117.63
Tip	186.07	116.85	0.00

Leg A	x	y	z
Joint 1	-80.00	2.50	50.00
Joint 2	-115.18	-6.25	50.00
Joint 3	-182.70	-23.03	117.95
Tip	-186.07	-23.87	0.00

Leg D	x	y	z
Joint 1	80.00	2.50	50.00
Joint 2	115.60	9.34	50.00
Joint 3	175.12	20.77	126.06
Tip	196.07	24.79	10.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	14.0	-26.4	-23.3	10.9
θ2 °	44.3	43.9	44.1	51.5
θ3 °	132.6	128.0	129.5	131.0

Step 12



Input (Step No.)

No. 12

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	170.00	50.00
Joint 2	-114.03	182.50	50.00
Joint 3	-179.48	206.54	117.79
Tip	-186.07	208.96	0.00

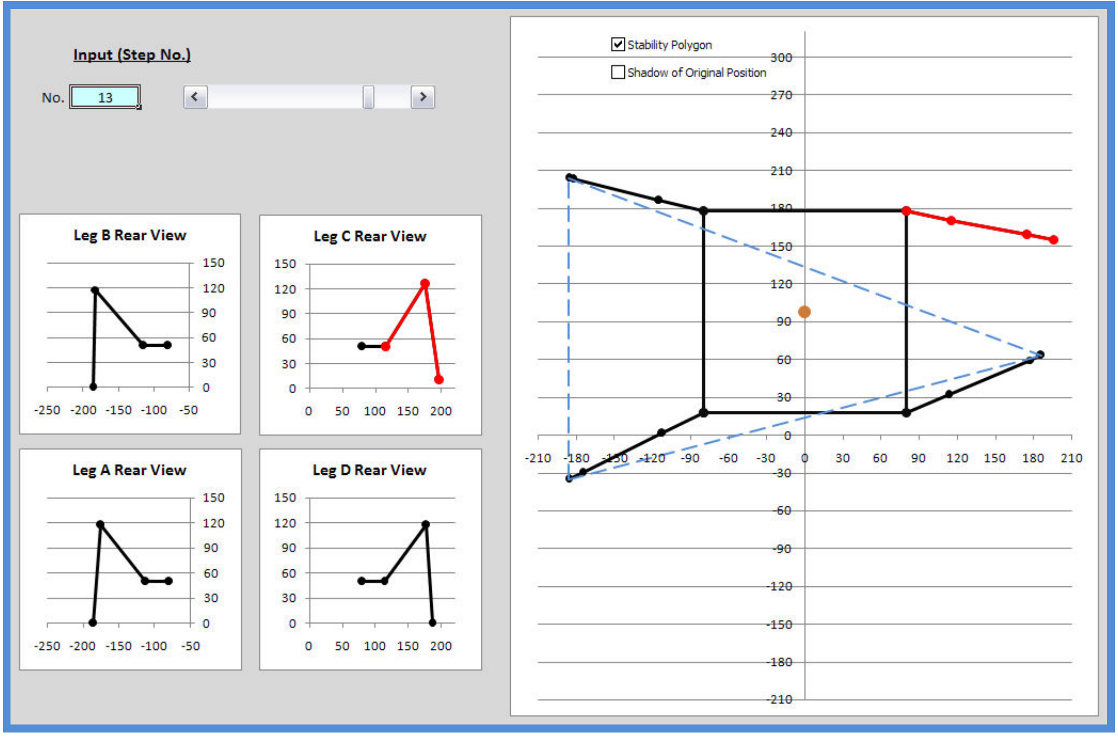
Leg C	x	y	z
Joint 1	80.00	170.00	50.00
Joint 2	111.55	152.15	50.00
Joint 3	172.88	117.46	117.02
Tip	186.07	110.00	0.00

Leg A	x	y	z
Joint 1	-80.00	10.00	50.00
Joint 2	-114.03	-2.50	50.00
Joint 3	-179.48	-26.54	117.79
Tip	-186.07	-28.96	0.00

Leg D	x	y	z
Joint 1	80.00	10.00	50.00
Joint 2	111.55	27.85	50.00
Joint 3	172.88	62.54	117.02
Tip	186.07	70.00	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	20.2	-20.2	-29.5	29.5
θ2 °	44.2	44.2	43.6	43.6
θ3 °	130.8	130.8	126.2	126.2

Step 13



Input (Step No.)

No. 13

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	177.50	50.00
Joint 2	-115.18	186.25	50.00
Joint 3	-182.70	203.03	117.95
Tip	-186.07	203.87	0.00

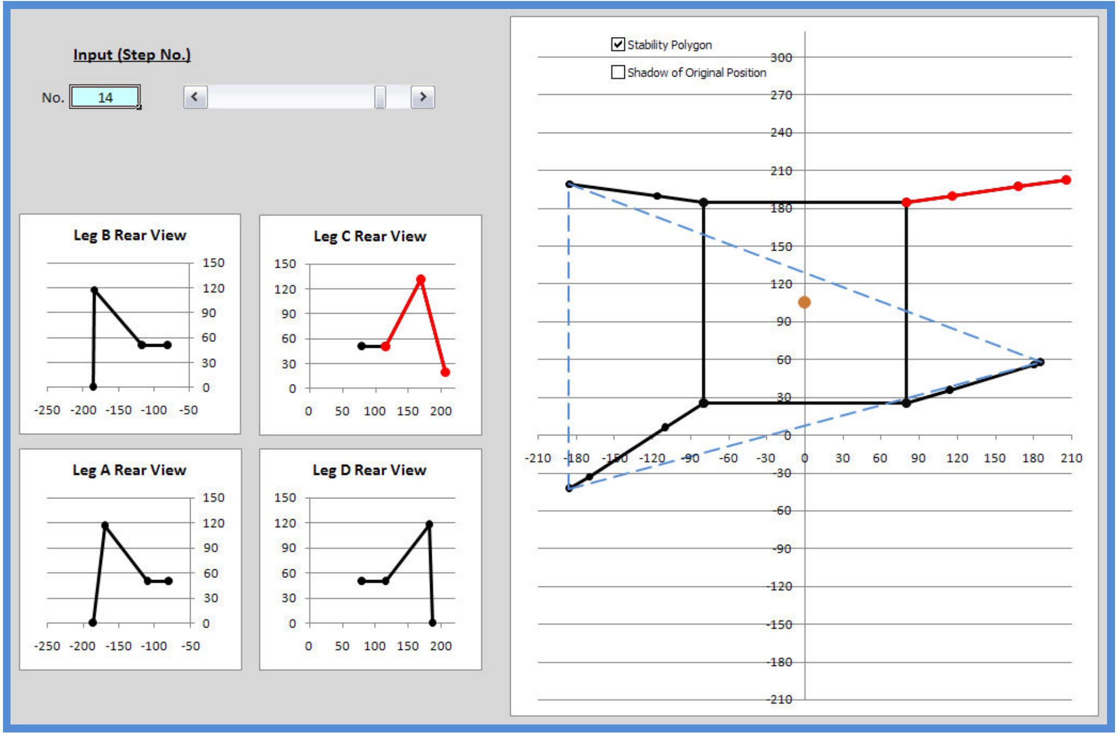
Leg C	x	y	z
Joint 1	80.00	177.50	50.00
Joint 2	115.60	170.66	50.00
Joint 3	175.12	159.23	126.06
Tip	196.07	155.21	10.00

Leg A	x	y	z
Joint 1	-80.00	17.50	50.00
Joint 2	-112.48	1.40	50.00
Joint 3	-175.30	-29.76	117.39
Tip	-186.07	-35.10	0.00

Leg D	x	y	z
Joint 1	80.00	17.50	50.00
Joint 2	113.30	31.83	50.00
Joint 3	177.49	59.46	117.63
Tip	186.07	63.15	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	26.4	-14.0	-10.9	23.3
θ2 °	43.9	44.3	51.5	44.1
θ3 °	128.0	132.6	131.0	129.5

Step 14



Input (Step No.)

No. 14

☑ Stability Polygon

□ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	185.00	50.00
Joint 2	-115.92	189.89	50.00
Joint 3	-184.81	199.27	117.99
Tip	-186.07	199.44	0.00

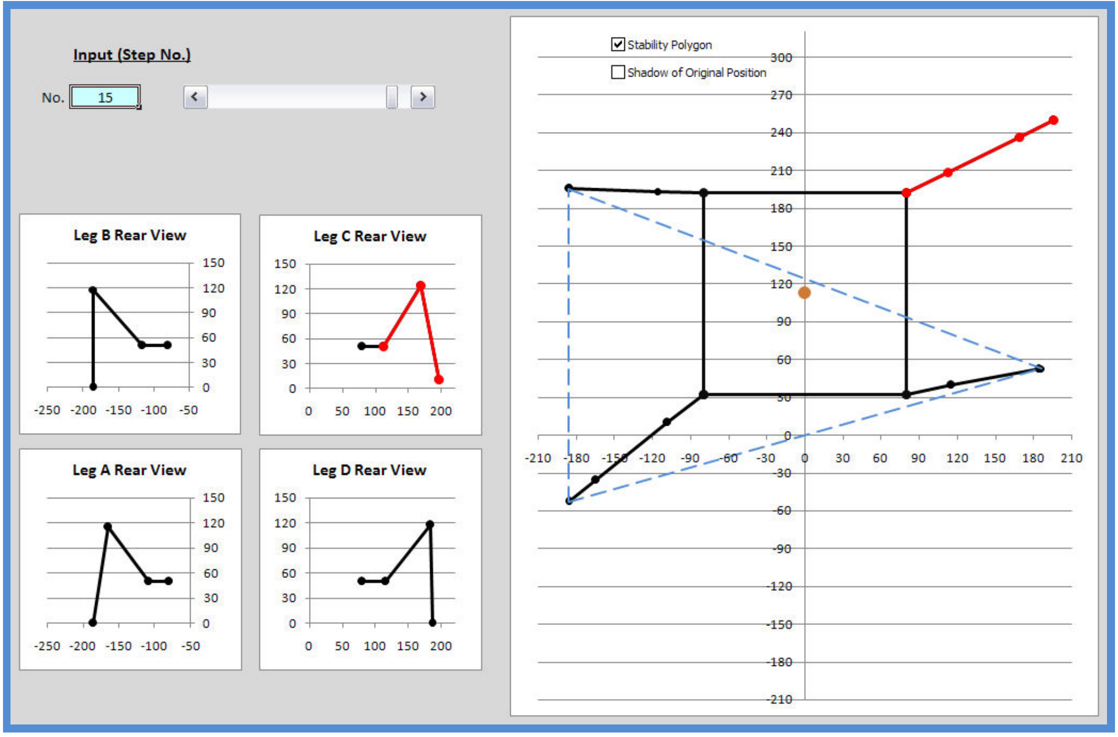
Leg C	x	y	z
Joint 1	80.00	185.00	50.00
Joint 2	115.92	189.89	50.00
Joint 3	168.25	197.01	131.66
Tip	206.07	202.16	20.00

Leg A	x	y	z
Joint 1	-80.00	25.00	50.00
Joint 2	-110.54	5.48	50.00
Joint 3	-170.33	-32.73	116.51
Tip	-186.07	-42.79	0.00

Leg D	x	y	z
Joint 1	80.00	25.00	50.00
Joint 2	114.65	35.65	50.00
Joint 3	181.21	56.10	117.89
Tip	186.07	57.59	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	32.6	-7.8	7.8	17.1
θ2 °	43.2	44.4	57.1	44.3
θ3 °	124.0	133.7	128.2	131.8

Step 15



Leg B	x	y	z
Joint 1	-80.00	192.50	50.00
Joint 2	-116.24	193.48	50.00
Joint 3	-185.74	195.35	118.00
Tip	-186.07	195.36	0.00

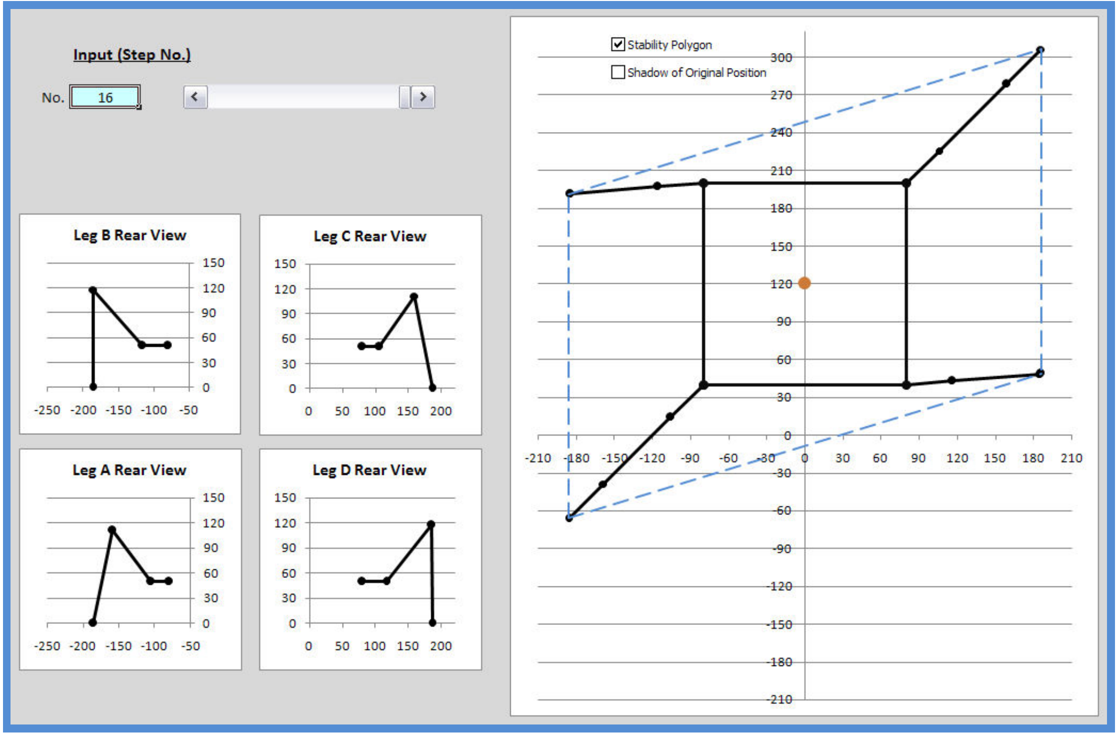
Leg C	x	y	z
Joint 1	80.00	192.50	50.00
Joint 2	112.48	208.60	50.00
Joint 3	168.95	236.61	124.05
Tip	196.07	250.06	10.00

Leg A	x	y	z
Joint 1	-80.00	32.50	50.00
Joint 2	-108.25	9.79	50.00
Joint 3	-164.78	-35.64	114.80
Tip	-186.07	-52.75	0.00

Leg D	x	y	z
Joint 1	80.00	32.50	50.00
Joint 2	115.60	39.34	50.00
Joint 3	183.90	52.45	117.98
Tip	186.07	52.87	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	38.8	-1.5	26.4	10.9
θ2 °	41.8	44.4	49.6	44.3
θ3 °	118.4	134.2	124.7	133.3

Step 16



No. 16

☑ Stability Polygon

☐ Shadow of Original Position

Leg B	x	y	z
Joint 1	-80.00	200.00	50.00
Joint 2	-116.13	197.05	50.00
Joint 3	-185.43	191.40	118.00
Tip	-186.07	191.35	0.00

Leg C	x	y	z
Joint 1	80.00	200.00	50.00
Joint 2	105.63	225.63	50.00
Joint 3	158.87	278.87	111.56
Tip	186.07	306.07	0.00

Leg A	x	y	z
Joint 1	-80.00	40.00	50.00
Joint 2	-105.63	14.37	50.00
Joint 3	-158.87	-38.87	111.56
Tip	-186.07	-66.07	0.00

Leg D	x	y	z
Joint 1	80.00	40.00	50.00
Joint 2	116.13	42.95	50.00
Joint 3	185.43	48.60	118.00
Tip	186.07	48.65	0.00

Angles	Leg A	Leg B	Leg C	Leg D
θ1 °	45.0	4.7	45.0	4.7
θ2 °	39.3	44.4	39.3	44.4
θ3 °	110.2	134.1	110.2	134.1

APPENDIX 16

Motor Controller Programs

Program of Leg A and Leg B

```
#include            <18F458.h>
#device             adc=8
#use                delay(clock=4000000)
#fuses              NOWDT,WDT128,RC_IO, NOPROTECT, NOOSCSEN, NOBROWNOUT, BORV20,
                    NOPUT, NOCPD, NOSTVREN, NODEBUG, NOLVP, NOWRT, NOWRTD, NOWRTB,
                    NOCPB, NOWRTC, NOEBTR, NOEBTRB

// Global_Variables
unsigned int16      duty_A1, duty_A2, duty_A3,
                    duty_B1, duty_B2, duty_B3;

unsigned int8       count, prev_step, curr_step;

signed int16        temp[6];

// Global_Constants (Look-Up Table)
//                                A1    A2    A3    B1    B2    B3
signed int16        LUT[17][6] = {1950, 2007, 1298, 1547, 1956, 1059,   // step_0
                                  1764, 1904, 1153, 1609, 1957, 1067,   // step_1
                                  1578, 1829, 1118, 1671, 1957, 1082,   // step_2
                                  1391, 1885, 1090, 1733, 1959, 1105,   // step_3
                                  1205, 1964, 1138, 1795, 1964, 1138,   // step_4
                                  1267, 1959, 1105, 1609, 1885, 1090,   // step_5
                                  1329, 1957, 1082, 1422, 1829, 1118,   // step_6
                                  1391, 1957, 1067, 1236, 1904, 1153,   // step_7
                                  1453, 1956, 1059, 1050, 2007, 1298,   // step_8
                                  1515, 1956, 1058, 1112, 1982, 1216,   // step_9
                                  1578, 1956, 1063, 1174, 1968, 1160,   // step_10
                                  1640, 1957, 1074, 1236, 1961, 1120,   // step_11
                                  1702, 1958, 1092, 1298, 1958, 1092,   // step_12
                                  1764, 1961, 1120, 1360, 1957, 1074,   // step_13
                                  1826, 1968, 1160, 1422, 1956, 1063,   // step_14
                                  1888, 1982, 1216, 1485, 1956, 1058,   // step_15
                                  1950, 2007, 1298, 1547, 1956, 1059 }; // step_16

// Define Leg_A
#define             joint_A1    PIN_D1
#define             joint_A2    PIN_D0
#define             joint_A3    PIN_C3

// Define Leg_B
#define             joint_B1    PIN_A0
#define             joint_B2    PIN_A1
#define             joint_B3    PIN_A2


// Leg_A
#int_RTCC
RTCC_isr()
{  set_RTCC(46035);                     // 65535 - 46035 = 19500counts
                                        // 19500counts X 1us per count = 0.0195sec
                                        // 1 / 0.0195sec = 51Hz

// Motor_A1
   output_high(joint_A1);
   delay_us(600);

   if((0x0258 <= duty_A1) && (duty_A1 <= 0x02EE))      // CCW: 76deg - 90deg
   {          delay_us(duty_A1-0x0258);
   }
```

```
else if((0x02EE < duty_A1) && (duty_A1 <= 0x03E8))     // CCW: 51deg - 75deg
{           delay_us(150);
            delay_us(duty_A1-0x02EE);
}

else if((0x03E8 < duty_A1) && (duty_A1 <= 0x04E2))     // CCW: 26deg - 50deg
{           delay_us(400);
            delay_us(duty_A1-0x03E8);
}

else if((0x04E2 < duty_A1) && (duty_A1 <= 0x05DC))     // CCW: 1deg - 25deg
{           delay_us(650);                               // 0deg == 0x05DC
            delay_us(duty_A1-0x04E2);
}

else if((0x05DC < duty_A1) && (duty_A1 <= 0x06D6))     // CW: 1deg - 25deg
{           delay_us(900);
            delay_us(duty_A1-0x05DC);
}

else if((0x06D6 < duty_A1) && (duty_A1 <= 0x07D0))     // CW: 26deg - 50deg
{           delay_us(1150);
            delay_us(duty_A1-0x06D6);
}

else if((0x07D0 < duty_A1) && (duty_A1 <= 0x08CA))     // CW: 51deg - 75deg
{           delay_us(1400);
            delay_us(duty_A1-0x07D0);
}

else if((0x08CA < duty_A1) && (duty_A1 <= 0x0960))     // CW: 76deg - 90deg
{           delay_us(1650);
            delay_us(duty_A1-0x08CA);
}

output_low(joint_A1);
// End of Motor_A1

// Motor_A2
output_high(joint_A2);
delay_us(600);

if((0x0258 <= duty_A2) && (duty_A2 <= 0x02EE))         // CCW: 76deg - 90deg
{           delay_us(duty_A2-0x0258);
}

else if((0x02EE < duty_A2) && (duty_A2 <= 0x03E8))     // CCW: 51deg - 75deg
{           delay_us(150);
            delay_us(duty_A2-0x02EE);
}

else if((0x03E8 < duty_A2) && (duty_A2 <= 0x04E2))     // CCW: 26deg - 50deg
{           delay_us(400);
            delay_us(duty_A2-0x03E8);
}

else if((0x04E2 < duty_A2) && (duty_A2 <= 0x05DC))     // CCW: 1deg - 25deg
{           delay_us(650);                               // 0deg == 0x05DC
            delay_us(duty_A2-0x04E2);
}

else if((0x05DC < duty_A2) && (duty_A2 <= 0x06D6))     // CW: 1deg - 25deg
{           delay_us(900);
            delay_us(duty_A2-0x05DC);
}

else if((0x06D6 < duty_A2) && (duty_A2 <= 0x07D0))     // CW: 26deg - 50deg
{           delay_us(1150);
            delay_us(duty_A2-0x06D6);
}
```

```
  else if((0x07D0 < duty_A2) && (duty_A2 <= 0x08CA))    // CW: 51deg - 75deg
  {           delay_us(1400);
              delay_us(duty_A2-0x07D0);
  }

  else if((0x08CA < duty_A2) && (duty_A2 <= 0x0960))    // CW: 76deg - 90deg
  {           delay_us(1650);
              delay_us(duty_A2-0x08CA);
  }

  output_low(joint_A2);
  // End of Motor_A2

   // Motor_A3
  output_high(joint_A3);
  delay_us(600);

  if((0x0258 <= duty_A3) && (duty_A3 <= 0x02EE))        // CCW: 76deg - 90deg
  {           delay_us(duty_A3-0x0258);
  }

  else if((0x02EE < duty_A3) && (duty_A3 <= 0x03E8))    // CCW: 51deg - 75deg
  {           delay_us(150);
              delay_us(duty_A3-0x02EE);
  }

  else if((0x03E8 < duty_A3) && (duty_A3 <= 0x04E2))    // CCW: 26deg - 50deg
  {            delay_us(400);
              delay_us(duty_A3-0x03E8);
  }

  else if((0x04E2 < duty_A3) && (duty_A3 <= 0x05DC))   // CCW: 1deg - 25deg
  {           delay_us(650);                            // 0deg == 0x05DC
              delay_us(duty_A3-0x04E2);
  }

  else if((0x05DC < duty_A3) && (duty_A3 <= 0x06D6))    // CW: 1deg - 25deg
  {            delay_us(900);
              delay_us(duty_A3-0x05DC);
  }

  else if((0x06D6 < duty_A3) && (duty_A3 <= 0x07D0))    // CW: 26deg - 50deg
  {            delay_us(1150);
              delay_us(duty_A3-0x06D6);
  }

  else if((0x07D0 < duty_A3) && (duty_A3 <= 0x08CA))    // CW: 51deg - 75deg
  {           delay_us(1400);
              delay_us(duty_A3-0x07D0);
  }

  else if((0x08CA < duty_A3) && (duty_A3 <= 0x0960))    // CW: 76deg - 90deg
  {           delay_us(1650);
              delay_us(duty_A3-0x08CA);
  }

  output_low(joint_A3);
  // End of Motor_A3
} // End of Leg_A


// Leg_B
#int_TIMER1
TIMER1_isr()
{  set_RTCC(46035);                          // 65535 - 46035 = 19500counts
                                             // 19500counts X 1us per count = 0.0195sec
                                             // 1 / 0.0195sec = 51Hz

// Motor_B1
  output_high(joint_B1);
  delay_us(600);
```

```
 if((0x0258 <= duty_B1) && (duty_B1 <= 0x02EE))          // CCW: 76deg - 90deg
 {           delay_us(duty_B1-0x0258);
 }

 else if((0x02EE < duty_B1) && (duty_B1 <= 0x03E8))     // CCW: 51deg - 75deg
 {           delay_us(150);
             delay_us(duty_B1-0x02EE);
 }

 else if((0x03E8 < duty_B1) && (duty_B1 <= 0x04E2))     // CCW: 26deg - 50deg
 {           delay_us(400);
             delay_us(duty_B1-0x03E8);
 }

 else if((0x04E2 < duty_B1) && (duty_B1 <= 0x05DC))     // CCW: 1deg - 25deg
 {           delay_us(650);                             // 0deg == 0x05DC
             delay_us(duty_B1-0x04E2);
 }

 else if((0x05DC < duty_B1) && (duty_B1 <= 0x06D6))     // CW: 1deg - 25deg
 {           delay_us(900);
             delay_us(duty_B1-0x05DC);
 }

 else if((0x06D6 < duty_B1) && (duty_B1 <= 0x07D0))     // CW: 26deg - 50deg
 {           delay_us(1150);
             delay_us(duty_B1-0x06D6);
 }

 else if((0x07D0 < duty_B1) && (duty_B1 <= 0x08CA))     // CW: 51deg - 75deg
 {           delay_us(1400);
             delay_us(duty_B1-0x07D0);
 }

 else if((0x08CA < duty_B1) && (duty_B1 <= 0x0960))     // CW: 76deg - 90deg
 {           delay_us(1650);
             delay_us(duty_B1-0x08CA);
 }

 output_low(joint_B1);
 // End of Motor_B1


// Motor_B2
 output_high(joint_B2);
 delay_us(600);

 if((0x0258 <= duty_B2) && (duty_B2 <= 0x02EE))          // CCW: 76deg - 90deg
 {           delay_us(duty_B2-0x0258);
 }

 else if((0x02EE < duty_B2) && (duty_B2 <= 0x03E8))     // CCW: 51deg - 75deg
 {           delay_us(150);
             delay_us(duty_B2-0x02EE);
 }

 else if((0x03E8 < duty_B2) && (duty_B2 <= 0x04E2))     // CCW: 26deg - 50deg
 {           delay_us(400);
             delay_us(duty_B2-0x03E8);
 }

 else if((0x04E2 < duty_B2) && (duty_B2 <= 0x05DC))     // CCW: 1deg - 25deg
 {           delay_us(650);                             // 0deg == 0x05DC
             delay_us(duty_B2-0x04E2);
 }

 else if((0x05DC < duty_B2) && (duty_B2 <= 0x06D6))     // CW: 1deg - 25deg
 {           delay_us(900);
             delay_us(duty_B2-0x05DC);
 }
```

```
    else if((0x06D6 < duty_B2) && (duty_B2 <= 0x07D0))     // CW: 26deg - 50deg
    {           delay_us(1150);
                delay_us(duty_B2-0x06D6);
    }

    else if((0x07D0 < duty_B2) && (duty_B2 <= 0x08CA))     // CW: 51deg - 75deg
    {           delay_us(1400);
                delay_us(duty_B2-0x07D0);
    }

    else if((0x08CA < duty_B2) && (duty_B2 <= 0x0960))     // CW: 76deg - 90deg
    {           delay_us(1650);
                delay_us(duty_B2-0x08CA);
    }

    output_low(joint_B2);
    // End of Motor_B2

    // Motor_B3
    output_high(joint_B3);
    delay_us(600);

    if((0x0258 <= duty_B3) && (duty_B3 <= 0x02EE))         // CCW: 76deg - 90deg
    {           delay_us(duty_B3-0x0258);
    }

    else if((0x02EE < duty_B3) && (duty_B3 <= 0x03E8))     // CCW: 51deg - 75deg
    {           delay_us(150);
                delay_us(duty_B3-0x02EE);
    }

    else if((0x03E8 < duty_B3) && (duty_B3 <= 0x04E2))     // CCW: 26deg - 50deg
    {            delay_us(400);
                delay_us(duty_B3-0x03E8);
    }

    else if((0x04E2 < duty_B3) && (duty_B3 <= 0x05DC))   // CCW: 1deg - 25deg
    {           delay_us(650);                           // 0deg == 0x05DC
                delay_us(duty_B3-0x04E2);
    }

    else if((0x05DC < duty_B3) && (duty_B3 <= 0x06D6))     // CW: 1deg - 25deg
    {            delay_us(900);
                delay_us(duty_B3-0x05DC);
    }

    else if((0x06D6 < duty_B3) && (duty_B3 <= 0x07D0))     // CW: 26deg - 50deg
    {            delay_us(1150);
                delay_us(duty_B3-0x06D6);
    }

    else if((0x07D0 < duty_B3) && (duty_B3 <= 0x08CA))     // CW: 51deg - 75deg
    {           delay_us(1400);
                delay_us(duty_B3-0x07D0);
    }

    else if((0x08CA < duty_B3) && (duty_B3 <= 0x0960))     // CW: 76deg - 90deg
    {           delay_us(1650);
                delay_us(duty_B3-0x08CA);
    }

    output_low(joint_B3);
    // End of Motor_B3
} // End of Leg_B


// Function: Steps_Counting
#int_TIMER3
TIMER3_isr()
{               set_timer3(40535);                  // 65535 - 25000 = 40535counts
                                                    // 25000counts X 2us = 50msec
```

```
  // Input_Switch
  if(input(PIN_B3) & 1)                              // Switch_ON
  {
    if(count < 10)                                   // 10counts X 50msec = 0.5sec
    {         count++;

      // Smoothing
      if(curr_step > prev_step)
      {
        temp[0] = LUT[prev_step][0] + ( LUT[curr_step][0] - LUT[prev_step][0] )*(count)/10,
        temp[1] = LUT[prev_step][1] + ( LUT[curr_step][1] - LUT[prev_step][1] )*(count)/10;
        temp[2] = LUT[prev_step][2] + ( LUT[curr_step][2] - LUT[prev_step][2] )*(count)/10,
        temp[3] = LUT[prev_step][3] + ( LUT[curr_step][3] - LUT[prev_step][3] )*(count)/10;
        temp[4] = LUT[prev_step][4] + ( LUT[curr_step][4] - LUT[prev_step][4] )*(count)/10,
        temp[5] = LUT[prev_step][5] + ( LUT[curr_step][5] - LUT[prev_step][5] )*(count)/10;
      } // End of Smoothing

      else
      {
        temp[0] = LUT[prev_step][0],
        temp[1] = LUT[prev_step][1],
        temp[2] = LUT[prev_step][2];
        temp[3] = LUT[prev_step][3],
        temp[4] = LUT[prev_step][4],
        temp[5] = LUT[prev_step][5];
      }
    }

    else
    {  count = 0;                        // Reset_count

      prev_step = curr_step;
      curr_step++;

      if(curr_step > 16)
      {  curr_step = 16;   //step = 0;
      }

      temp[0] = LUT[prev_step][0],
      temp[1] = LUT[prev_step][1],
      temp[2] = LUT[prev_step][2];
      temp[3] = LUT[prev_step][3],
      temp[4] = LUT[prev_step][4],
      temp[5] = LUT[prev_step][5];
    }
  } // End of Switch_ON

  else                                               // Switch_OFF
  {
    curr_step = 0;
    temp[0] = LUT[curr_step][0],
    temp[1] = LUT[curr_step][1],
    temp[2] = LUT[curr_step][2];
    temp[3] = LUT[curr_step][3],
    temp[4] = LUT[curr_step][4],
    temp[5] = LUT[curr_step][5];
  } // End of Input_Switch

  // Assign temp[] to duty_XX
  duty_A1 = temp[0],
  duty_A2 = temp[1],
  duty_A3 = temp[2];
  duty_B1 = temp[3],
  duty_B2 = temp[4],
  duty_B3 = temp[5];

}  // End of Function: Steps_Counting
```

```
// Function: Main
void main()
{
  setup_adc_ports(NO_ANALOGS);
  setup_adc(ADC_OFF);
  setup_psp(PSP_DISABLED);
  setup_spi(FALSE);
  setup_wdt(WDT_OFF);
  setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
  setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
  setup_timer_2(T2_DISABLED,0,1);
  setup_timer_3(T3_EXTERNAL|T3_DIV_BY_8);
  setup_comparator(NC_NC_NC_NC);
  enable_interrupts(INT_RTCC);
  enable_interrupts(INT_TIMER1);
  enable_interrupts(INT_TIMER3);
  enable_interrupts(GLOBAL);

  // Local_Variables
  count            = 0;
  prev_step        = 0;
  curr_step        = 0;

  // Loop_Forever
  while(1)
  {
  }
} // End of Main
```

Program of Leg C and Leg D

```
#include            <18F458.h>
#device             adc=8
#use                delay(clock=4000000)
#fuses              NOWDT,WDT128,RC_IO, NOPROTECT, NOOSCSEN, NOBROWNOUT, BORV20,
                    NOPUT, NOCPD, NOSTVREN, NODEBUG, NOLVP, NOWRT, NOWRTD, NOWRTB,
                    NOCPB, NOWRTC, NOEBTR, NOEBTRB

// Global_Variables
unsigned int16      duty_C1, duty_C2, duty_C3,
                    duty_D1, duty_D2, duty_D3;

unsigned int8       count, prev_step, curr_step;

signed int16        temp[6];

// Global_Constants (Look-Up Table)
//                                    C1    C2    C3    D1    D2    D3
signed int16        LUT[17][6] =    {1950,  993, 1702, 1547, 1044, 1941,   // step_0
                                     1888, 1018, 1784, 1485, 1044, 1942,   // step_1
                                     1826, 1032, 1840, 1422, 1044, 1937,   // step_2
                                     1764, 1039, 1880, 1360, 1043, 1926,   // step_3
                                     1702, 1042, 1908, 1298, 1042, 1908,   // step_4
                                     1640, 1043, 1926, 1236, 1039, 1880,   // step_5
                                     1578, 1044, 1937, 1174, 1032, 1840,   // step_6
                                     1515, 1044, 1942, 1112, 1018, 1784,   // step_7
                                     1453, 1044, 1941, 1050,  993, 1702,   // step_8
                                     1391, 1043, 1933, 1236, 1096, 1847,   // step_9
                                     1329, 1043, 1918, 1422, 1171, 1882,   // step_10
                                     1267, 1041, 1895, 1609, 1115, 1910,   // step_11
                                     1205, 1036, 1862, 1795, 1036, 1862,   // step_12
                                     1391, 1115, 1910, 1733, 1041, 1895,   // step_13
                                     1578, 1171, 1882, 1671, 1043, 1918,   // step_14
                                     1764, 1096, 1847, 1609, 1043, 1933,   // step_15
                                     1950,  993, 1702, 1547, 1044, 1941 }; // step_16
// Define Leg_C
#define             joint_C1   PIN_B7
#define             joint_C2   PIN_B6
#define             joint_C3   PIN_B5

// Define Leg_D
#define             joint_D1   PIN_D2
#define             joint_D2   PIN_D3
#define             joint_D3   PIN_C4

// Leg_C
#int_RTCC
RTCC_isr()
{  set_RTCC(46035);                    // 65535 - 46035 = 19500counts
                                       // 19500counts X 1us per count = 0.0195sec
                                       // 1 / 0.0195sec = 51Hz

 // Motor_C1
   output_high(joint_C1);
   delay_us(600);

   if((0x0258 <= duty_C1) && (duty_C1 <= 0x02EE))        // CCW: 76deg - 90deg
   {          delay_us(duty_C1-0x0258);
   }

   else if((0x02EE < duty_C1) && (duty_C1 <= 0x03E8))    // CCW: 51deg - 75deg
   {          delay_us(150);
              delay_us(duty_C1-0x02EE);
   }

   else if((0x03E8 < duty_C1) && (duty_C1 <= 0x04E2))    // CCW: 26deg - 50deg
   {          delay_us(400);
              delay_us(duty_C1-0x03E8);
   }
```

```
else if((0x04E2 < duty_C1) && (duty_C1 <= 0x05DC))    // CCW: 1deg - 25deg
{           delay_us(650);                            // 0deg == 0x05DC
            delay_us(duty_C1-0x04E2);
}

else if((0x05DC < duty_C1) && (duty_C1 <= 0x06D6))     // CW: 1deg - 25deg
{           delay_us(900);
            delay_us(duty_C1-0x05DC);
}

else if((0x06D6 < duty_C1) && (duty_C1 <= 0x07D0))     // CW: 26deg - 50deg
{           delay_us(1150);
            delay_us(duty_C1-0x06D6);
}

else if((0x07D0 < duty_C1) && (duty_C1 <= 0x08CA))     // CW: 51deg - 75deg
{           delay_us(1400);
            delay_us(duty_C1-0x07D0);
}

else if((0x08CA < duty_C1) && (duty_C1 <= 0x0960))     // CW: 76deg - 90deg
{           delay_us(1650);
            delay_us(duty_C1-0x08CA);
}

output_low(joint_C1);
// End of Motor_C1

// Motor_C2
output_high(joint_C2);
delay_us(600);

if((0x0258 <= duty_C2) && (duty_C2 <= 0x02EE))         // CCW: 76deg - 90deg
{           delay_us(duty_C2-0x0258);
}

else if((0x02EE < duty_C2) && (duty_C2 <= 0x03E8))     // CCW: 51deg - 75deg
{           delay_us(150);
            delay_us(duty_C2-0x02EE);
}

else if((0x03E8 < duty_C2) && (duty_C2 <= 0x04E2))     // CCW: 26deg - 50deg
{           delay_us(400);
            delay_us(duty_C2-0x03E8);
}

else if((0x04E2 < duty_C2) && (duty_C2 <= 0x05DC))    // CCW: 1deg - 25deg
{           delay_us(650);                            // 0deg == 0x05DC
            delay_us(duty_C2-0x04E2);
}

else if((0x05DC < duty_C2) && (duty_C2 <= 0x06D6))     // CW: 1deg - 25deg
{           delay_us(900);
            delay_us(duty_C2-0x05DC);
}

else if((0x06D6 < duty_C2) && (duty_C2 <= 0x07D0))     // CW: 26deg - 50deg
{           delay_us(1150);
            delay_us(duty_C2-0x06D6);
}

else if((0x07D0 < duty_C2) && (duty_C2 <= 0x08CA))     // CW: 51deg - 75deg
{           delay_us(1400);
            delay_us(duty_C2-0x07D0);
}

else if((0x08CA < duty_C2) && (duty_C2 <= 0x0960))     // CW: 76deg - 90deg
{           delay_us(1650);
            delay_us(duty_C2-0x08CA);
}

output_low(joint_C2);
// End of Motor_C2
```

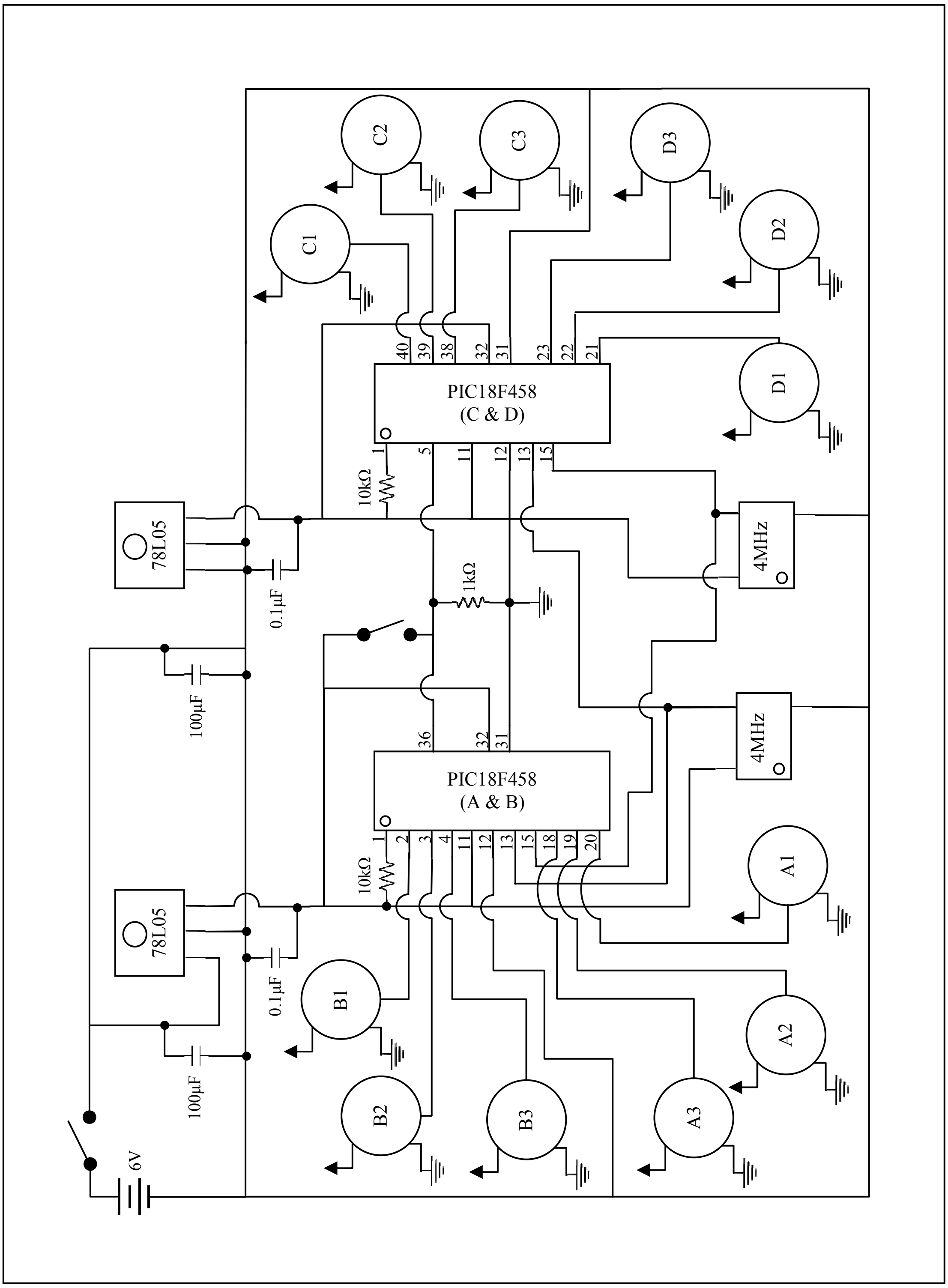
```
  // Motor_C3
  output_high(joint_C3);
  delay_us(600);

  if((0x0258 <= duty_C3) && (duty_C3 <= 0x02EE))        // CCW: 76deg - 90deg
  {          delay_us(duty_C3-0x0258);
  }

  else if((0x02EE < duty_C3) && (duty_C3 <= 0x03E8))     // CCW: 51deg - 75deg
  {          delay_us(150);
             delay_us(duty_C3-0x02EE);
  }

  else if((0x03E8 < duty_C3) && (duty_C3 <= 0x04E2))     // CCW: 26deg - 50deg
  {           delay_us(400);
             delay_us(duty_C3-0x03E8);
  }

  else if((0x04E2 < duty_C3) && (duty_C3 <= 0x05DC))   // CCW: 1deg - 25deg
  {          delay_us(650);                             // 0deg == 0x05DC
             delay_us(duty_C3-0x04E2);
  }

  else if((0x05DC < duty_C3) && (duty_C3 <= 0x06D6))    // CW: 1deg - 25deg
  {           delay_us(900);
             delay_us(duty_C3-0x05DC);
  }

  else if((0x06D6 < duty_C3) && (duty_C3 <= 0x07D0))    // CW: 26deg - 50deg
  {           delay_us(1150);
             delay_us(duty_C3-0x06D6);
  }

  else if((0x07D0 < duty_C3) && (duty_C3 <= 0x08CA))    // CW: 51deg - 75deg
  {          delay_us(1400);
             delay_us(duty_C3-0x07D0);
  }

  else if((0x08CA < duty_C3) && (duty_C3 <= 0x0960))    // CW: 76deg - 90deg
  {          delay_us(1650);
             delay_us(duty_C3-0x08CA);
  }

  output_low(joint_C3);
  // End of Motor_C3
} // End of Leg_C

// Leg_D
#int_TIMER1
TIMER1_isr()
{  set_RTCC(46035);                     // 65535 - 46035 = 19500counts
                                        // 19500counts X 1us per count = 0.0195sec
                                        // 1 / 0.0195sec = 51Hz

 // Motor_D1
  output_high(joint_D1);
  delay_us(600);

  if((0x0258 <= duty_D1) && (duty_D1 <= 0x02EE))        // CCW: 76deg - 90deg
  {          delay_us(duty_D1-0x0258);
  }

  else if((0x02EE < duty_D1) && (duty_D1 <= 0x03E8))    // CCW: 51deg - 75deg
  {          delay_us(150);
             delay_us(duty_D1-0x02EE);
  }

  else if((0x03E8 < duty_D1) && (duty_D1 <= 0x04E2))    // CCW: 26deg - 50deg
  {          delay_us(400);
             delay_us(duty_D1-0x03E8);
  }
```

```
else if((0x04E2 < duty_D1) && (duty_D1 <= 0x05DC))    // CCW: 1deg - 25deg
{           delay_us(650);                              // 0deg == 0x05DC
            delay_us(duty_D1-0x04E2);
}

else if((0x05DC < duty_D1) && (duty_D1 <= 0x06D6))     // CW: 1deg - 25deg
{           delay_us(900);
            delay_us(duty_D1-0x05DC);
}

else if((0x06D6 < duty_D1) && (duty_D1 <= 0x07D0))    // CW: 26deg - 50deg
{           delay_us(1150);
            delay_us(duty_D1-0x06D6);
}

else if((0x07D0 < duty_D1) && (duty_D1 <= 0x08CA))     // CW: 51deg - 75deg
{           delay_us(1400);
            delay_us(duty_D1-0x07D0);
}

else if((0x08CA < duty_D1) && (duty_D1 <= 0x0960))    // CW: 76deg - 90deg
{           delay_us(1650);
            delay_us(duty_D1-0x08CA);
}

output_low(joint_D1);
// End of Motor_D1

// Motor_D2
output_high(joint_D2);
delay_us(600);

if((0x0258 <= duty_D2) && (duty_D2 <= 0x02EE))        // CCW: 76deg - 90deg
{           delay_us(duty_D2-0x0258);
}

else if((0x02EE < duty_D2) && (duty_D2 <= 0x03E8))    // CCW: 51deg - 75deg
{           delay_us(150);
            delay_us(duty_D2-0x02EE);
}

else if((0x03E8 < duty_D2) && (duty_D2 <= 0x04E2))    // CCW: 26deg - 50deg
{           delay_us(400);
            delay_us(duty_D2-0x03E8);
}

else if((0x04E2 < duty_D2) && (duty_D2 <= 0x05DC))   // CCW: 1deg - 25deg
{           delay_us(650);                             // 0deg == 0x05DC
            delay_us(duty_D2-0x04E2);
}

else if((0x05DC < duty_D2) && (duty_D2 <= 0x06D6))     // CW: 1deg - 25deg
{           delay_us(900);
            delay_us(duty_D2-0x05DC);
}

else if((0x06D6 < duty_D2) && (duty_D2 <= 0x07D0))    // CW: 26deg - 50deg
{           delay_us(1150);
            delay_us(duty_D2-0x06D6);
}

else if((0x07D0 < duty_D2) && (duty_D2 <= 0x08CA))     // CW: 51deg - 75deg
{           delay_us(1400);
            delay_us(duty_D2-0x07D0);
}

else if((0x08CA < duty_D2) && (duty_D2 <= 0x0960))    // CW: 76deg - 90deg
{           delay_us(1650);
            delay_us(duty_D2-0x08CA);
}

output_low(joint_D2);
// End of Motor_D2
```

```
    // Motor_D3
    output_high(joint_D3);
    delay_us(600);

    if((0x0258 <= duty_D3) && (duty_D3 <= 0x02EE))        // CCW: 76deg - 90deg
    {           delay_us(duty_D3-0x0258);
    }

    else if((0x02EE < duty_D3) && (duty_D3 <= 0x03E8))     // CCW: 51deg - 75deg
    {           delay_us(150);
                delay_us(duty_D3-0x02EE);
    }

    else if((0x03E8 < duty_D3) && (duty_D3 <= 0x04E2))     // CCW: 26deg - 50deg
    {            delay_us(400);
                delay_us(duty_D3-0x03E8);
    }

    else if((0x04E2 < duty_D3) && (duty_D3 <= 0x05DC))   // CCW: 1deg - 25deg
    {           delay_us(650);                             // 0deg == 0x05DC
                delay_us(duty_D3-0x04E2);
    }

    else if((0x05DC < duty_D3) && (duty_D3 <= 0x06D6))     // CW: 1deg - 25deg
    {            delay_us(900);
                delay_us(duty_D3-0x05DC);
    }

    else if((0x06D6 < duty_D3) && (duty_D3 <= 0x07D0))     // CW: 26deg - 50deg
    {            delay_us(1150);
                delay_us(duty_D3-0x06D6);
    }

    else if((0x07D0 < duty_D3) && (duty_D3 <= 0x08CA))     // CW: 51deg - 75deg
    {           delay_us(1400);
                delay_us(duty_D3-0x07D0);
    }

    else if((0x08CA < duty_D3) && (duty_D3 <= 0x0960))     // CW: 76deg - 90deg
    {           delay_us(1650);
                delay_us(duty_D3-0x08CA);
    }

    output_low(joint_D3);
    // End of Motor_D3
  } // End of Leg_D


  // Function: Steps_Counting
  #int_TIMER3
  TIMER3_isr()
  {             set_timer3(40535);                  // 65535 - 25000 = 40535counts
                                                    // 25000counts X 2us = 50msec

    // Input_Switch
    if(input(PIN_A3) & 1)                           // Switch_ON
    {
      if(count < 10)                                // 10counts X 50msec = 0.5sec
      {         count++;

        // Smoothing
        if(curr_step > prev_step)
        {
          temp[0] = LUT[prev_step][0] + ( LUT[curr_step][0] - LUT[prev_step][0] )*(count)/10,
          temp[1] = LUT[prev_step][1] + ( LUT[curr_step][1] - LUT[prev_step][1] )*(count)/10;
          temp[2] = LUT[prev_step][2] + ( LUT[curr_step][2] - LUT[prev_step][2] )*(count)/10,
          temp[3] = LUT[prev_step][3] + ( LUT[curr_step][3] - LUT[prev_step][3] )*(count)/10;
          temp[4] = LUT[prev_step][4] + ( LUT[curr_step][4] - LUT[prev_step][4] )*(count)/10,
          temp[5] = LUT[prev_step][5] + ( LUT[curr_step][5] - LUT[prev_step][5] )*(count)/10;
        } // End of Smoothing
        else
        {
          temp[0] = LUT[prev_step][0],
          temp[1] = LUT[prev_step][1],
          temp[2] = LUT[prev_step][2];
          temp[3] = LUT[prev_step][3],
          temp[4] = LUT[prev_step][4],
          temp[5] = LUT[prev_step][5];
        }
      }
      else
      {  count = 0;                    // Reset_count

        prev_step = curr_step;
        curr_step++;

        if(curr_step > 16)
        {  curr_step = 16;   //step = 0;
        }
        temp[0] = LUT[prev_step][0],
        temp[1] = LUT[prev_step][1],
        temp[2] = LUT[prev_step][2];
        temp[3] = LUT[prev_step][3],
        temp[4] = LUT[prev_step][4],
        temp[5] = LUT[prev_step][5];
      }
    } // End of Switch_ON
    else                                                  // Switch_OFF
    {
      curr_step = 0;
      temp[0] = LUT[curr_step][0],
      temp[1] = LUT[curr_step][1],
      temp[2] = LUT[curr_step][2];
      temp[3] = LUT[curr_step][3],
      temp[4] = LUT[curr_step][4],
      temp[5] = LUT[curr_step][5];
    } // End of Input_Switch

    // Assign temp[] to duty_XX
    duty_C1 = temp[0],
    duty_C2 = temp[1],
    duty_C3 = temp[2];
    duty_D1 = temp[3],
    duty_D2 = temp[4],
    duty_D3 = temp[5];
  } // End of Function: Steps_Counting

  // Function: Main
  void main()
  { setup_adc_ports(NO_ANALOGS);
    setup_adc(ADC_OFF);
    setup_psp(PSP_DISABLED);
    setup_spi(FALSE);
    setup_wdt(WDT_OFF);
    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_1);
    setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
    setup_timer_2(T2_DISABLED,0,1);
    setup_timer_3(T3_EXTERNAL|T3_DIV_BY_8);
    setup_comparator(NC_NC_NC_NC);
    enable_interrupts(INT_RTCC);
    enable_interrupts(INT_TIMER1);
    enable_interrupts(INT_TIMER3);
    enable_interrupts(GLOBAL);
    // Local_Variables
    count              = 0;
    prev_step          = 0;
    curr_step          = 0;

    // Loop_Forever
    while(1)
    {
    }
  } // End of Main
```

APPENDIX 17

Motor Controller Wiring Diagram


C1
C2
C3
D1
D2
D3
40
39
38
32
31
23
22
21
PIC18F458
(C & D)
1
5
11
12
13
15
10kΩ
78L05
0.1µF
1kΩ
4MHz
100µF
36
32
31
PIC18F458
(A & B)
4MHz
1
2
3
4
11
12
13
15
18
19
20
10kΩ
78L05
0.1µF
A1
A2
A3
B1
B2
B3
100µF
6V