The George W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology
Atlanta, Georgia 30332
NASA/USRA
University Advanced Design Program

Three Legged Walking Mobile Platform
Kinematic & Dynamic
Analysis and Simulation

June 1988

Gary V. McMurray
Brice K. MacLaren
A THREE LEGGED WALKER

Georgia Institute of Technology
Atlanta, Georgia

The three legged walker is proposed as a mobile work platform for numerous tasks associated with Lunar Base site preparation and construction. It is seen as one of several forms of surface transportation, each of which will be best suited for its respective tasks.

Utilizing the principle of dynamic stability and taking advantage of the Moon's Gravity, it appears to be capable of walking in any radial direction and rotating about a point. Typical curved path walking could involve some combination of the radial and rotational movements.

Comprised mainly of a body, six actuators, and six moving parts, it is mechanically quite simple. Each leg connects to the body at a hip joint and has a femur, a knee joint, and a tibia that terminates at a foot.

Also capable of enabling or enhancing the dexterity of a series of implements, the walker concept provides a mechanically simple and weight efficient means of drilling, digging, mining, and transporting cargo, and performing other like tasks.

A proof of principle machine has demonstrated the feasibility of the walking concept.
# Table of Contents

I. Abstract ........................................ 1  
II. Introduction ................................. 1  
III. Motion ........................................ 3  
   A. Lean ........................................ 3  
   B. Leap ........................................ 5  
   C. Crutch Walk ................................ 5  
   D. Squat ....................................... 8  
   E. Pivoting .................................... 8  
   F. Self-Righting Mode ....................... 9  
IV. Dynamics and Kinematics .............. 10  
   A. Jump Mode ................................. 10  
   B. Lean Mode ................................. 15  
V. Dynamic Simulation ...................... 21  
   A. Computer Model ......................... 21  
   B. Results .................................. 23  
VI. Proof of Principle Model ............. 30  
VII. Conclusion ................................. 31  
Appendix A - Figures  
Appendix B - SKIT_3-D Graphic Simulation, Figures, and Program Listing  
Appendix C - SKITTER Dynamics and Kinematics Program Listing
Introduction

The Georgia Institute of Technology has been involved in the design of machinery for the construction of the Lunar Base for several years. Because of the unique and stringent constraints imposed upon any piece of machinery that will operate in the Lunar environment, a three-legged walking robot, called SKITTER, is being designed and developed. While SKITTER has been initially designed for Lunar applications, it is not solely limited to the moon. Some terrestrial applications include hazardous environments, military reconnaissance, underwater operations, etc. The purpose of this paper is to discuss and detail some of the initial work leading to the development of the theory of a three-legged walker (gaits, modes of operation, kinematics, and dynamics) and the proof of principle model.

The task of designing automated machinery for the Lunar environment is very difficult. Besides the intense temperatures and lack of an atmosphere, the one-sixth gravity complicates the task of moving machinery and cargo. The reduced gravity, while making any payload lighter, also reduces the normal force at the ground to such a low point that it becomes extremely difficult to be able to do even the basic task of scraping soil at a construction site. Even though the weight has been reduced by one-sixth, the inertia has not been similarly reduced. Thus, it is not a sufficient solution to the problem to add more mass to the vehicle to generate the required normal force because the power requirements for acceleration and deceleration would then rise sharply. Since any construction machine must be able to traverse all terrain that it might encounter, the decision to use legged locomotion over wheeled vehicles was justified.

SKITTER is a very simple device from a mechanical point of view. It consists of only six actuators, six moving parts, and a central body (Figs. 1 and 2). The legs are located radially
through the centerline of the body at 120 degrees apart from the other. The upper part of the leg, or femur, connects to the body at a hip joint. Connected at the other end of the femur is the lower leg, or tibia, which terminates as a foot at its end. To move the legs, two actuators per leg are used. The first actuator rotates the femur about the hinge line formed by the union of the femur and the central body, while the second actuator rotates the tibia about the hinge line created by the union of the femur and tibia. In this way, each leg operates in its own plane. The central body serves as a connect point for various implements that may be attached to the walker and as host for the electronic hardware and power supply.

This mechanical simplicity does, however, have its disadvantage in the fact that more complex controls are needed for machine stability. For stability in motion, the main difference between a three-legged walker and the other walking devices like the Odetics and Ohio State walking machines is that the Georgia Tech walker depends on dynamic stability to maintain its motion. It must, and does, operate routinely with less than three legs having contact with the ground at any particular moment. As the device pushes off from the ground with its legs, the center of mass undergoes a horizontal and vertical motion. This differs from the previously mentioned walkers dramatically in that they strive to constantly maintain a level motion of the center of mass. While it does take energy to move the center of mass vertically, that energy is recovered when the body and leg are brought back into contact with the ground with no energy expenditure. Since the walker is not statically stable at all times in this motion, the controls complexity increases dramatically. This situation is similar to the unipod and bipods being developed at Carnegie Mellon and Clemson Universities, however with two major distinctions. The first is that while the tripod is not statically stable at all times while in motion, the device can always return to a statically stable position by simply allowing the leg or legs to return to the ground provided that the center of mass of the device is still located inside of the triangle formed by the foot projections on the ground. The second major difference is related to the first one in that only very small movements of the legs are needed to generate motion of the
platform. By taking small steps, the danger of tipping and energy consumption is minimized.

This motion of pushing off from the ground so that the foot actually leaves the ground is also significant from another point of view. That it is possible to establish a rocking motion of the walker such that the inertia and momentum from the restoration of the leg back to the ground, aids in the pushing off from the ground of the other legs. As will be discussed later in this paper, this type of motion is very similar to that of a man on crutches.

Motion

Mechanical simplicity is a primary design constraint for SKITTER. Although other walkers incorporate many complex linkages and bearings in their design, SKITTER utilizes only six actuators (two per leg) and six hinges (two per leg) to generate motion (fig. 1 and 2). The femur actuator changes the angular position of the femur relative to the central body, and the tibia actuator changes the angular position of the tibia relative to the femur. By coordinating the position and velocities of the actuators, a variety of platform positions and motions is achieved.

Lean

SKITTER's basic mode of operation is to reorient its central body or "lean" by reconfiguring the legs while always maintaining three fixed points of contact with the surface. To understand the lean sequence as well as the other modes of operation described later in this section, a fixed axis (X-Y-Z) is established such that the walking surface lies in the X-Z plane and the positive Y axis is normal to the surface following the right hand rule. SKITTER is oriented such that the motions of leg A are confined to the X-Y plane and the feet of leg B and leg C construct a line parallel to the Z axis known as pivot line A. A second axis (x-y-z)
may be established such that the x-z plane lies in the mid-plane of the central body with the positive x axis in the direction of leg A and the positive z axis parallel to and in the same direction as the Z axis. The positive y axis is normal to the mid-plane and follows the right hand rule. Finally, a third axis (x'-y'-z') may also be established on the femur at the femur A - tibia A hinge line such that the positive z' axis is parallel and in the same direction as the positive Z axis (fig 3).

One possible example of the lean mode sequence is shown in fig 4. Starting with SKITTER at a static equilibrium position known as the 90-90 configuration (the femur-tibia and tibia-surface angles are both 90 degrees), femur A slowly rotates cw about the z axis and tibia A slowly rotates ccw about the z' axis such that foot A never leaves its initial contact point with the surface. The motion of femur A and tibia A is analogous to the motion of a crank and coupler of a slider crank mechanism discussed in detail in the kinematics section of this paper. The movements of leg A cause the central body and other two legs of SKITTER to rotate about pivot line A. As femur A and tibia A reach a new desired position, the central body has rotated and translated from its initial position with respect to the X-Y-Z reference frame.

One interesting motion of the central body is its ability to translate along its local y axis at an obtainable platform configuration. This can be accomplished by actuating each leg such that all three body-femur joints have a velocity vector parallel to the y axis and of equal magnitude. Therefore, as a drill rig platform, SKITTER eliminates the need for angular positioning and vertical feed mechanisms by leaning to the correct orientation and then raising and lowering itself along the drill string path by a series of coordinated actuator movements. The operation is completed with SKITTER's feet never losing contact with the ground. Also, the platform is able to achieve a position such that the mid-plane of the central body is parallel to the plane of its feet by simply making the body-femur angle and the femur-tibia angle of all three legs equal respectively. The work volume which encloses all of the possible orientations of the
central body is limited to the range of the actuators and the physical characteristics of the body and leg components.

Leap

One method for maneuvering around obstacles which might impair the movement of SKITTER such as small rocks or ditches is to "leap" over them. SKITTER simply reorients its central body such that the y axis lies in the intended direction of travel. All three legs move such that the body-femur joints of all three legs have a velocity vector of equal magnitude and parallel to the y axis and supply a sufficient downward force to make SKITTER leap. With increases in control logic and proper frame design, the magnitude of the leap increases giving SKITTER the ability to achieve larger distances, and thus imitating the "skip walk" used by the astronauts on the lunar missions. One advantage of the leap mode is that the magnitude of directions in which SKITTER could translate is limited only by the possible orientations of the central body; therefore, with proper design, true omni-directional motion can be obtained.

Crutch Walk

SKITTER's crutch walk mode for translational motion differs dramatically from most current walker designs which usually move one or more appendages while keeping at least three points of surface contact at all times. SKITTER, on the other hand, tries to capitalize off of its inertial characteristics and dynamic stability to propel itself forward. One inherent fact of a three legged platform such as SKITTER is that it will always be statically stable as long as all three feet are in contact with the surface and its center of gravity is positioned over the triangle formed by the feet. However, if one of the legs loses contact with the surface, the platform becomes statically unstable and starts rotating due to gravity about the pivot line constructed by the feet of the other two legs. By combining this fact with the lean motion,
SKITTER can be made to translate over a surface similar to a person walking with crutches.

For example, starting at the 90-90 equilibrium position, the femur and tibia of leg A begin the slider crank motion described in the kinematics section of this paper (fig 5). The central body starts to rotate about pivot line A as in the lean mode; however, this time, femur A and tibia A have acquired enough angular acceleration to supply a sufficient force at the foot and consequently a sufficient torque about pivot line A to cause foot A to lose contact with the surface (i.e. foot A pushes off from the surface fig 6). The entire platform continues to rotate about pivot line A until SKITTER's potential energy equals the kinetic energy imparted by leg A as it left the surface. At this point, the entire platform rotates about pivot line A in the opposite direction due to gravity. While leg A is away from the surface, femur A and tibia A rotate into a new configuration causing foot A to swing towards the central body (fig 6). As leg A comes back into contact with the surface, the central body is in a new orientation and foot A has translated to a new location on the surface in relation to the X-Y-Z reference frame (fig 7).

The next stage has leg B and leg C moving identically in their respective planes of motion. legs B & C reconfigure as shown in figure 7 causing a rotation of the central body in the X-Y plane about foot A. During their reconfiguration, legs B & C acquire adequate angular acceleration to supply a sufficient force at foot B and foot C, and consequently adequate torque about foot A, to cause the feet to lose contact with the surface (fig 8). Again, the platform will continue to rotate in the X-Y plane about foot A until its potential energy equals the kinetic energy imparted by legs B & C as they left the surface. At that time, the platform begins to rotate in the opposite direction due to gravity. While away from the surface all three legs reconfigure to their original 90-90 starting configuration (fig 8). As foot B and foot C reestablish contact with the surface, it is seen that the feet are in a new location and that the center of gravity has translated (fig 9). It is important to point out that if the roles of
leg A and legs B & C are interchanged (Reversed Crutch Walk mode) then a translation in the opposite direction occurs giving six radial directions of translation without a required rotation of the platform.

Surprisingly, it has been found by analysis that it takes little energy to have a leg push off from the surface with adequate force to give the leg time to reconfigure into a new position. Similarly, only a small rotation of the platform about the pivot line is needed to give adequate space for reconfiguration of the leg; therefore, the chances of the platform tipping over are small.

One important benefit which arises out of the Crutch Walk motion is a decrease in the energy input to the system as the platform gains momentum while it walks. With an increase in the gait of the crutch walk sequence and the proper control strategy, SKITTER is able to achieve a stable rocking motion. Just as a person who is walking quickly on crutches uses his momentum to swing himself forward, SKITTER uses its momentum to propel itself forward. Therefore, the horsepower to maintain the rocking motion is small since the energy input to the system only has to account for the losses in the system due to SKITTER contacting the surface.

Slopes can be negotiated quite easily using either the Crutch Walk or Reverse Crutch Walk mode with the requirement that the force vector due to gravity acting through the cg of SKITTER always intersect the triangle formed by the three feet. This requirement insures that SKITTER will not over turn and can always revert to a statically stable position. The grade of slope that SKITTER can effectively negotiate is primarily determined by the femur and tibia dimensions which determine the size of the triangle. A larger foot print triangle results in a larger margin of safety from over turning and therefore a larger grade of slope can be negotiated. The platform is able to walk up, down or tact a slope by assuming an optimum
nominal position (i.e. taking the largest step possible without overturning) and proceeding with one of the sequences described above.

Squat

SKITTER has the ability to lower its central body by having each leg repeat the sequence of pushing off of the surface, reconfiguring so that the foot swings away from the central body and landing on the surface to reestablish static equilibrium. If the sequence is carried through enough iterations, the central body of SKITTER would come to rest on the surface with the legs extended outward (fig. 10). This particular position is extremely advantageous if the platform is being used in conjunction with a lifting device such as a crane. In the squat mode, the legs form outriggers to counter the weight of the cargo being lifted and eliminate the need for counter weights or other stability mechanisms.

Pivoting

Although the platform has six radial directions for translation, there will be situations that will require for the platform to rotate about the surface Y axis. SKITTER is capable of two different pivoting modes. The foot pivoting mode allows the platform to pivot around one foot while the complex pivot mode allows SKITTER to swing one foot through an arc in the surface X-Z plane.

In the foot pivoting mode, Skitter pushes leg A off the surface, and while in the air, leg B reconfigures resulting in a torque about foot C. The platform will pivot around foot C, and as leg A contacts the surface, SKITTER will once again be statically stable. Since the hinge lines of the platform dictate a 120 degree interval between the planes of motion of the legs, the foot pivoting mode would also require that foot B either slide in a arc about foot C or be away
from the surface for the rotation to occur.

In the complex pivoting mode, SKITTER pushes leg A off the surface, and while in the air, leg B increases its body-femur angle while leg C decreases its body-femur angle. The reconfiguring of the legs in this manner will cause the platform to undergo rotations around the surface Y & X axis causing leg A to swing in a arc in the surface X-Z plane while both leg A and leg B remain fixed to the ground. Once leg A contacts the ground again, the platform becomes statically stable. If the motion is carried through for all three legs, SKITTER achieves a net rotation about its cg. Unlike the foot pivoting mode, the complex mode does not require foot B to slide or leave the surface for the rotation to occur.

Self Righting Mode

If, for some reason, the platform tripped or fell during one of the modes of operation, it has the capability of righting itself since the legs have a range of motion extending above and below the mid-plane of the central body (fig 11). As an extreme example, if SKITTER tripped and landed completely upside down on the surface, the platform could tuck two legs in toward the central body while the third leg pushed down on the ground to flip the platform over to the correct orientation (fig. 12). The resulting motion would simulate a person summersaulting and landing on his feet. This unique fault tolerant capability of SKITTER makes the platform a valuable remote field robot.

The movements just discussed were achieved by utilizing the inertia characteristics of SKITTER in conjunction with the coordinated actions of the six linear actuators. A direct relation between movement complexity and control complexity is apparent; however, the movements discussed be realized by current control strategies and devices.
DYNAMICS and KINEMATICS

Of the many possible combinations of motions of the femur and tibia joints, only two possible combinations of motions exist such that the feet do not slide on the ground. The first motion is a linear movement of the center of mass as in the jump mode of operation. To accomplish this, the femur and tibia joints must combine their motions to produce a linear motion at each hinge line of the femur and the body. To model this linear motion of the center of mass, each leg is modeled as an offset slider crank mechanism. If it is desired for the walker to actually leave the ground, then the device must supply enough force that the walker has sufficient velocity at the end of the leg movements to leave the ground, or jump. The derivation of this model is developed here and the results of that model presented later.

The second method of motion for the walker is one that produces a rotational displacement of the center of mass relative to a pivot line, or the lean motion. This lean motion is also the fundamental motion for the crutch walk discussed earlier. The only difference between the motions of the femur and tibia joints in the lean configuration and the crutch walk is that at the end of the crutch walk motion, the body has enough angular velocity to allow the foot to leave the ground and this allows the leg to reconfigure while off the ground. To model this rotational motion of the body, a four-bar linkage is constructed where the four links are the ground, the tibia, the femur, and a link that is composed of the rigid structure of the body and the other two legs. The final link is connected to the hinge of the femur and body of the leg that is moving and terminates at the pivot line formed by the two feet that do not move. Derivation of the kinematic and dynamic model is discussed in this section and the results of the computer simulation is discussed in a latter section.
The first motion to be discussed is the jump motion. This motion can be divided into to
distinct phases. The first phase is the acceleration of the body in the local vertical axis to a
prescribed velocity such that during phase two, the jump phase, the body is off the ground and
it begins to decelerate under the force of gravity. Since linear motion is required, an offset
slider crank mechanism is used to simulate the motion of the legs (Fig. 13a). As a
simplification of the mathematical model, the problem was inverted such that the body of
SKITTER was considered to be ground and the foot was constrained to move in the linear
fashion. Thus, the femur is considered to be the crank and the foot is the slider. A derivation
of that model is presented here along with a measure of the forces, torques, and angular
velocities needed to produce this motion.

To conduct this analysis, several parameters must be defined by the user of the program
developed to model this motion. The first parameter is the vertical distance, H, desired for
SKITTER to jump (Fig. 13c). The second parameter is the vertical distance, D, that the legs
are allowed to displace to accelerate the body to the desired velocity (Fig. 13b). This velocity,
V_0, is obtained using the conservation of potential and kinetic energy theory where the final
height is the jump distance, H, and the initial velocity is V. This results in,

\[ V_0 = (2gH)^{1/2} \]

In this equation, g represents the gravitational constant. For simplicity, constant linear
acceleration, A_0, is maintained at the center of mass for the acceleration phase. Thus, the
acceleration necessary to accelerate the body to the desired velocity, V_0, over the linear
distance, D, is,

\[ A_0 = \frac{V_0^2}{2D} \]
Given the initial joint angles of the leg (the angle between the local x axis and the femur centerline, $\theta_1$, and the angle between the femur and tibia centerlines, $\theta_2$), the other critical angle for the dynamic and kinematic analysis is the angle between the tibia and the vertical axis, $\phi$ (Fig. 13a). This angle is determined from the joint angles to be,

$$\theta_2 - \theta_1 - \pi/2 = \phi$$

The next step is to derive the angular velocities for the femur and tibia links. The angular velocity equations are written using the standard equations for finding the velocity of a link. The equations are written from both ends of the link (i.e., one equation relates the velocity of point B to the ground, A, and the other relates the velocity of point B to the foot, point C, where the velocity at point C is a known parameter). The two equations are,

$$V_b = V_a + \omega_{ab} \times R_{ab}$$

$$V_b = V_c + \omega_{bc} \times R_{cb}$$

When the above equations are evaluated for this particular geometry, the following equation is derived,

$$(-\omega_{ab} L \sin(\theta_1)) \mathbf{i} = (-\omega_{ab} L \cos(\theta_1)) \mathbf{i} = (-\omega_{bc} L \cos(\phi)) \mathbf{i} = (V_0 + \omega_{bc} L \sin(\phi)) \mathbf{i}$$

By comparing the $\mathbf{i}$ and $\mathbf{j}$ terms of the above equations, then the magnitude of the angular velocities can be derived,

$$\omega_{ab} = V_0 \cos(\phi)/L(\cos(\theta_1 + \phi))$$
\[
\omega_{bc} = \omega_{ab}\sin(\theta_1)/\cos(\phi)
\]

Like the velocity equations presented earlier, acceleration equations can be written in a similar manner.

\[
\begin{align*}
\Delta_b &= \Delta_a + \omega_{ab} R_{ab} + \omega_{ab} x (\omega_{ab} x R_{ab}) \\
\Delta_b &= \Delta_c + \omega_{bc} R_{cb} + \omega_{bc} x (\omega_{bc} x R_{cb})
\end{align*}
\]

Evaluating the above equations for the geometry of this problem yields,

\[
\begin{align*}
(-\omega_{ab}\sin(\theta_1) - \omega_{ab}\sin^2(\theta_1)) \downarrow + (-\omega_{ab}\cos(\theta_1) + \omega_{ab}\sin(\theta_1)) \downarrow = \\
(-\omega_{bc}\cos(\phi) + \omega_{bc}\sin^2(\phi)) \downarrow + (-A_0 - \omega_{bc}\sin(\phi) - \omega_{bc}\cos(\phi)) \downarrow
\end{align*}
\]

Solving the above equations properly for the magnitude of the angular accelerations gives,

\[
\begin{align*}
\alpha_{bc} &= -A_0 \sin(\theta_1) - \omega_{bc}^2 \sin(\theta_1) - \omega_{bc}^2 \cos^2(\phi) - \omega_{ab}^2 \sin(\theta_1) - \omega_{ab}^2 \cos^2(\phi) - \omega_{bc}^2 L^2 / \cos(\phi + \theta_1)) \\
\alpha_{ab} &= (\omega_{bc} L\cos(\phi) - \omega_{bc}^2 L\sin(\phi) - \omega_{ab}^2 L\sin(\theta_1)) / \sin(\theta_1)
\end{align*}
\]

For a complete force and torque analysis of the leg motion, it is necessary to obtain the linear acceleration of the center of mass for the tibia, \( \dot{X}_{cg} \) and \( \dot{Y}_{cg} \). Using the acceleration equation for the center of mass relative to the foot gives,

\[
\Delta_{cg} = \Delta_c + \omega_{bc} R_{ccg} + \omega_{bc} x (\omega_{bc} x R_{ccg})
\]
Evaluating this equation for the geometry yields,

\[ A_{cg} = (-\alpha_{bc} L \cos(\theta)/2 + \omega_{bc}^2 L \sin(\theta)/2) \downarrow + (-A_0 - \alpha_{bc} L \sin(\theta)/2 - \omega_{bc}^2 L \cos(\theta)/2) \downarrow \]

Which leads to,

\[ X_{cg} = -\alpha_{bc} L \cos(\theta)/2 + \omega_{bc}^2 L \sin(\theta)/2 \]

\[ Y_{cg} = -A_0 - \alpha_{bc} L \sin(\theta)/2 - \omega_{bc}^2 L \cos(\theta)/2 \]

Summing the forces for the tibia (note that the force P is the reaction force from the ground due to the weight of SKITTER) enables a determination of the reaction forces at point B,

\[ \Sigma F_x = m_T X_{cg} = F_{bx} \]

\[ \Sigma F_y = m_T Y_{cg} = P - F_{by} - m_T g \]

Solving for the reaction forces,

\[ F_{bx} = m_T X_{cg} \]

\[ F_{by} = m_T Y_{cg} + m_T g - P \]

The summation of the moments about points A and B are,

\[ \Sigma M_b = I_T \alpha_{bc} = PL \sin(\theta) - m_T g L \sin(\theta)/2 - T_t \]
\[ \Sigma M_a = I_f \alpha_{ab} = F_{by}L\cos(\theta_1) - F_{bx}L\sin(\theta_1) - m_TgL\cos(\theta_1)/2 - T_f \]

Thus, the torque needed about the femur and tibia are,

\[ T_t = I_t \alpha_{bc} - P L\sin(\phi) + m_TgL\sin(\phi)/2 \]

\[ T_f = I_f \alpha_{ab} + F_{bx}L\sin(\theta_1) + m_TgL\cos(\theta_1)/2 - F_{by}L\cos(\theta_1) \]

The calculations for the mass moments of inertia are approximations based upon a rectangular cross section of the leg. The variable "a" is equal to the length of the leg and the variable "b" is equal to the width of the cross section of the leg.

\[ I_x = m_{leg} (a^2 + b^2) / 12 \]

From the above equations, all the variables are known except for the torques. Therefore, the torque needed at either the femur or tibia joints is known for any given position of the legs.

These equations were then implemented in a computer program to evaluate the angular velocity, torque, and horse power requirements for each joint of the leg as it attempts to make the leg jump the desired distance.

**LEAN MOTION**

As stated earlier, the lean motion is the foundation of the crutch walk and provides dexterity to the platform. The kinematic model for this mode of operation is a four-bar
linkage where the four links are (Fig. 14a),

- Tibia
- Femur
- Rigid structure consisting of everything but links one and two
- Ground

The lean motion can also be divided into two separate phases of motion. The first is the acceleration phase where the femur and tibia move in such a fashion as to impart to the center of mass a prescribed angular velocity. This angular velocity is sufficient to allow the foot to leave the ground after the forces have ceased to be applied to the joints.

As in the jump mode analysis, it is necessary to input two parameters into the program to allow the remainder of the variables to be set. For the lean mode, it is necessary for the user to define the acceleration angle, \( \varphi \), which is the angle that the center of mass of SKITTER is to undergo to obtain the necessary angular velocity (Fig. 14b). The other input parameter is the angular displacement, \( \theta \), that the user wants the center of mass to undergo after the foot leaves the ground (Fig. 14b).

With these input parameters, an initial angular velocity must be calculated so that the center of mass will undergo the desired amount of rotation. This initial angular velocity, \( \omega_0 \), is calculated from the conservation of potential and kinetic energy theory where the final angular velocity is zero and the final height of the center of mass is \( H_2 \), where \( H_2 \) is defined as (Fig. 14b):

\[
H_2 = rsin(\varphi+\theta)
\]
Where \( r \) is defined as the constant perpendicular distance from the center of mass of SKITTER to the pivot line formed by the stationery feet. Thus, the initial angular velocity must equal:

\[
\omega_0 = \left(\frac{2mgH_2}{I}\right)^{1/2}
\]

For this equation, the \( m \) terms refers to the mass of SKITTER and the \( I \) term is the moment of inertia for SKITTER about the pivot line.

As with the jump motion, a constant angular acceleration, \( \alpha_0 \), is assumed over the acceleration angle for the center of mass. This acceleration results in the desired angular velocity, \( \omega_0 \), at the end of the acceleration angle, \( \psi \).

\[
\alpha_0 = \frac{\omega_0^2}{(2\psi)}
\]

For this motion, there are a number of geometric parameters that must be summarized for the following analysis to be clear. First, the joint angles for the different links are designated as follows (Fig. 14a):

- \( \theta_1 \): angle between the tibia link and the ground
- \( \theta_2 \): angle between the femur link and horizontal at the femur-tibia joint
- \( \theta_3 \): angle between the rigid body of SKITTER and the ground.
As in the jump motion, all of these angles are measured in the right hand sense. The letters designate the various joints in this analysis. Point A is the hip joint, point B is the knee joint, point C represents the contact point between the ground and the tibia, and point D is the contact point for the line that is perpendicular to the pivot line and contains the center of mass.

With this variable definition, the following equations can be written for the velocity of the points on the links:

\[
\begin{align*}
\dot{Y}_b &= \dot{Y}_c + \omega_{bc} \times R_{cb} \\
\dot{Y}_a &= \dot{Y}_b + \omega_{ab} \times R_{ba} \\
\dot{Y}_d &= \dot{Y}_c + \omega_{ad} \times R_{da}
\end{align*}
\]

The above equations are evaluated using the following conditions, \(Y_c = Y_d = 0\). With this above condition and the fact that \(\omega_{ad}\) is the input velocity following a prescribed acceleration profile, then the remaining angular velocity terms can be calculated by solving the above equations for the scalar magnitudes of the angular velocities:

\[
\begin{align*}
\omega_{ab} &= R\omega_{ad} \frac{\sin(\theta_1 - \theta_3)}{(L\sin(\theta_1 - \theta_2))} \\
\omega_{bc} &= (\omega_{ad} R \sin(\theta_3) - \omega_{ab} L \sin(\theta_2)) / L \sin(\theta_1)
\end{align*}
\]

As with the velocity equations, similar equations for the angular acceleration can be written.

\[
\begin{align*}
\ddot{A}_b &= \ddot{A}_c + \omega_{bc} \times R_{cb} + \omega_{bc} \times (\omega_{bc} \times R_{cb})
\end{align*}
\]
Again applying the boundary conditions of $\Delta_c = \Delta_d = 0$ and the fact that the angular acceleration, $\alpha_{ad}$, is the $\alpha_0$ calculated earlier, the following magnitudes for the angular accelerations may be derived.

$$\alpha_{ab} = (R\alpha_{ad}\sin(\theta_1 - \theta_3) - R\omega_{ad}^2\cos(\theta_1 - \theta_3) + L\omega_{ab}^2\cos(\theta_2 - \theta_1) + L\omega_{bc}^2) / \sin(\theta_1 - \theta_2)$$

$$\alpha_{bc} = (\alpha_{ad}R\sin(\theta_2) + \omega_{ad}^2R\cos(\theta_2) - \omega_{bc}^2L\cos(\theta_1) - \alpha_{ab}L\sin(\theta_2) - \omega_{ab}^2L\cos(\theta_2)) / \sin(\theta_1)$$

Up to this point, the angular velocity and angular accelerations have been calculated for the various links. To complete the analysis for the forces and torques necessary to achieve the input motion, the linear acceleration of the center of mass for the femur, tibia, and rigid body must be calculated. Since all the velocities and accelerations are already known, this is very simple.

$$X_{T, cg} = A_{b, x}/2$$

$$Y_{T, cg} = A_{b, y}/2$$

$$X_{F, cg} = A_{bx} - (\alpha_{ab}L\sin(\theta_2) - \omega_{ab}^2L\cos(\theta_2))/2$$

$$Y_{F, cg} = A_{bx} + (\alpha_{ab}L\cos(\theta_2) - \omega_{ab}^2L\sin(\theta_2))/2$$
Now the free body diagrams can be written for this system. Unfortunately, the system is a coupled one, which means that the input torque, $T$, about the femur can not be easily solved for without solving for eight other variables at the same time. These additional variables are the reaction forces at the joints and ground. Thus, the equations can be best solved by putting the equations into matrix form and then solving them using Gaussian elimination with pivoting.

The form of the equations to be solved is as follows:

$$
\begin{bmatrix}
0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & \text{Lsin}(\theta) - \text{Lcos}(\theta) & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\
\text{Lsin}(\theta_2) - \text{Lcos}(\theta_2) & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
\text{Rsin}(\theta_3) - \text{Rcos}(\theta_3) & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
F_{a,x} \\
F_{a,y} \\
F_{b,x} \\
F_{b,y} \\
F_{c,x} \\
F_{c,y} \\
F_{d,x} \\
F_{d,y} \\
T
\end{bmatrix} =
\begin{bmatrix}
m_{T}X_{T,cs} \\
m_{T}Y_{T,cs} \\
\text{L}_c \text{b} + m_{L} \text{L}
\text{cos}(\theta_1)/2 \\
\text{L}_{F}X_{F,cs} \\
\text{L}_{F}Y_{F,cs} \\
\text{L}_{c} \text{b} + m_{L} \text{L}
\text{cos}(\theta_2)/2 \\
m_{\text{Body}}X_{\text{Body},cs} \\
m_{\text{Body}}Y_{\text{Body},cs} \\
m_{\text{Body}} \text{L}
\text{cos}(\theta_3)/2
\end{bmatrix}
$$

From the solution of this matrix for the variable $T$, the input torque required to have SKITTER lean and lift off from the ground can be calculated.
Dynamic Simulation of the Motion of SKITTER

To simulate the motion of SKITTER, the dynamic and kinematic equations were developed for the basic motions, the jump and the lean motion, and then implemented in a computer program. These motions do not cover the broad scope of motions possible, especially all the possible motions resulting from the non-symmetric configurations of the legs. However, these two motions do represent the basic modes of operation for the machine (the other gaits are a combination of these two motions). Also included in this analysis is an actuator sizing routines which allows the user to determine if a given actuator (rotary or linear) can supply the necessary torque and speed.

The programs developed are written in a general format to allow the user to vary the physical parameters of SKITTER as well as modify its performance parameters. These performance parameters include the distance that the walker will jump in the air, the distance that the legs accelerate through before they leave the ground, and the actuator specifications (torque and velocity limitations). The physical parameters that can be varied on the model include all of the actuator attach points, the length of the femur and tibia, the weight of the femur and tibia (and thus its inertia), the weight of SKITTER, the gravity, and the type of actuator (rotary or linear).

From the dynamic analysis of the motion of SKITTER, a maximum torque and angular velocity about the hip and knee joints of the walker are calculated. If the type of actuator is a rotary one, then these values are compared against the input specifications for the actuator to determine if they will suffice. For a linear actuator, the moment arm about each joint must be calculated for that instant in time. The reason for this is that as the leg undergoes its motion, a linear actuator will not maintain a constant perpendicular
distance from the joint. Thus, whether a specified actuator will provide the prescribed motion is a function of two variables: torque required and perpendicular distance. To determine if the actuator will work, the two worst cases must be compared the actuator specifications. If the actuator is able to provide the necessary linear force and velocity for the two cases of maximum torque and minimum perpendicular distance, then the actuator will work.

**Jump Motion**

The jump motion is described in detail in the another section of this paper and will not be redefined here. The dynamic model for this motion is one that provides linear motion of the hip joint (and thus the central body) as compared to the foot. Thus, the kinematic model for this motion is an off-set slider crank mechanism where the input is from the hip joint and the knee joint is a passive joint.

The solution to this kinematic problem was implemented in a computer program in order to solve for the angular velocities and accelerations of each joint. These values were very important in order that the dynamic problem could be solved completely. To accomplish this, the Newtonian Force equations were derived for the linkage so that the torque about each of the joints could be solved for as a function of its position.

**Lean Motion**

As with the jump motion, the lean motion is described in detail elsewhere in this paper. For this problem, the motion is modeled as a four-bar linkage where the joints are comprised of:

1) Tibia

2) Femur
3) Rigid structure consisting of everything but links 1 and 2 (the rest of SKITTER)

4) Ground

The kinematic solution to this problem is also well known but with one small difference: the input to the system undergoes a constant acceleration. While this is not a drastic change in the problem, it does considerably complicate the solution. With this accomplished, the next task was to derive the dynamic solution to determine the input torque needed to generate the angular velocities calculated by the kinematic equations as necessary to move SKITTER. Again using Newtonian mechanics and the angular accelerations and velocities from the kinematic problem, a system matrix of rank nine was derived that had to be solved to determine the input torque. The system matrix, while not triangular, was solvable using Gaussian elimination with row pivoting to obtain a solution to the input torque needed at the hip joint to provide the desired motion.

Results

Presented below are the results of the numerical calculations and simulations performed under this contract. All of the number represent the power requirements of SKITTER while operating on earth and with a weight of three hundred pounds.

Jump Motion

Sample data presented below is from the computer simulation of the dynamics of SKITTER. The family of curves shown below is for a variety of jump heights and acceleration distances. The physical parameters for the SKITTER model were determined from the envisioned SKITTER II model. The values shown are the requirements for each
leg as it attempts to jump. The total system requires three times as much torque and horse power to jump the distance desired.

The first plot shows a family of curves for the input torque at the hip joint at various jump distances and acceleration distances.

The following plot shows required maximum angular velocity at the hip joint to accelerate SKITTER to jump various heights through different acceleration distances.
As can be seen from these plots, the requirements to jump are not as strenuous as originally thought. Sizing actuators to meet these specifications is not difficult and a vendor has already been identified that can meet these requirements. The Helac Corporation makes a series of planetary rotary actuators that are capable of supplying 4300 inch-pounds of torque with a full 360 degrees of rotation in the joint, while weighing only 24 pounds. These actuators will be ideal choice for use on SKITTER II.

**Lean Motion**

For the lean motion, a family of curves was generated for various angles of acceleration (this is the angle that SKITTER accelerates through till it reaches the desired angular velocity to lift off from the ground) and ground clearances (the distance that the
foot is from the ground at its maximum point). It is important to note that this simulation is for the case where one leg is providing the force to rotate SKITTER about the other two legs. In the crutch walk motion described earlier, it is also desired for the other two legs to push off from the ground and rotate about the stationary third leg during part of its motion. For this case, the angular velocity about each of the moving legs hip joint is the same whether one or two legs is pushing, but the torque can be divided between the two legs. Thus, the angular velocity requirements stay the same for this motion, but the torque requirements are divided between the two legs.

The first plot illustrates the family of curves relating the maximum input torque at the hip joint for a variety of acceleration angles and ground clearances (the distance that the foot of SKITTER leaves the ground during the rotation motion).
The next plot relates the maximum angular velocity at the hip joint and a family of acceleration angles and ground clearance.
The data presented here has been checked in various manners. For the lean motion, graphical techniques were used to verify the angular velocities and accelerations generated by the computer program. Since these numbers are then used to determine the input torque, this number is believed to be correct also.

Future Work

The work completed under this contract to develop kinematic and dynamic equations for the motions of SKITTER has been completed. Actuator sizing programs have been developed so that the designer can vary the size of the structural members and optimize the power consumption for a given size actuator. The joint angles from the jump and lean motions can also be written to a file so that the graphical simulation program can display
the motions of SKITTER in 3-D. The next step in this process is to improve the computer models and incorporate in the program a control algorithm and inefficiencies in the power transmission. This will give the most accurate computer simulation of what the actual SKITTER II will be like when it eventually is built. Again, by using the computer simulations, the designer is able to optimize the design before any hardware is built. This allows the best prototype to be built.
PROOF OF PRINCIPLE MODEL

To test the theory on the motion of a three-legged walker, a proof of principle working model was constructed. The walker was a one-tenth scale model of the conceptual design of the Lunar model and its purpose was only to obtain some translatory motion. This model, which weighed approximately eighty-five pounds and was completely self-contained, was demonstrated to NASA on several occasions where it fulfilled its intended goal and also demonstrated several other of the modes of operation discussed earlier.

The SKITTER proof of principle model was pneumatically actuated for cost effective reasons. A small scuba tank was attached to the underside of the model to serve as a high pressure reservoir of air. With a pneumatic actuation system, each actuator was only able to move the joint into two discrete positions, either the actuator was all the way in or all the way out (as compared to hydraulic or electromechanical actuators that can reach an infinite number of positions over its stroke length). Since the original design called for two actuators per leg, this results in four positions of the foot. To improve on this, the design for the prototype was modified such that each joint consisted of two actuators, four actuators per leg, and the actuators were connected by a free floating member. This gave each joint four possible positions that it could obtain and thus the foot could obtain sixteen different positions.

To control the actuation of the joints, a small computer was located aboard the model. Since no usable position or velocity feedback can be obtained from a pneumatic actuator, the computer was programed with a series of commands that controlled the actuators and therefore the walker could demonstrate the modes of operations discussed earlier.

There are a number of fundamental differences between the proof of principle model and the SKITTER proposed for use on the moon. The first difference is that the prototype operated outside of its intended environment on the moon where the gravity is one-sixth's of
the earth's gravity. Thus, the power requirements were much higher on the prototype as compared to a comparable SKITTER on the moon. The second difference is that the prototype used a discrete actuation system that allows the feet to be in only sixteen discrete motions. Of those positions, only certain movements from these discrete positions allowed the prototype to move as designed. The final difference was that no control strategy could be implemented to provide smooth motion of the legs because of the pneumatic system's lack of useful feedback.

The fact that a successful prototype was built and demonstrated with these negative factors inhibiting its performance, shows that the idea is feasible and much easier to implement than originally thought. With the addition of servo-actuators, the motion can only be improved, but the fundamental concepts on the modes of operation of a three-legged walker have already been proved correct.

**CONCLUSION**

With a successful proof of principle model developed, the Georgia Institute of Technology is continuing the development of the three-legged walker, SKITTER. To further a complete understanding of the dynamics and kinematics of the walker, computer simulation is being written to incorporate the equations of motion and display graphically SKITTER as it moves. Once this work has been completed, a control strategy and hardware (actuators and sensors) will be incorporated into the computer model to provide a realistic simulation of the next generation prototype. This model can then be evaluated and modified by the user before any hardware is actually built.

The next version of SKITTER, or SKITTER II, will have servo-actuators at the joints to allow feedback of the position and velocity of the joint so that the motion of the legs can be accurately controlled throughout the range of their motions. This new model will be capable of all the gaits and modes of operation described previously, but it will have much greater range
of motion and a much smoother motion. This next generation SKITTER model will also be compared against other walking and wheeled vehicles for overall efficiency, as well as the accuracy of the computer models.
Appendix A

Figures
SKITTER 2
WALKING MOBILE PLATFORM

10 - BODY
20 - LEG
30 - FEMUR
40 - TIBIA
42 - FOOT
50 - ACTUATOR
52 - HIP JOINT
54 - KNEE JOINT
CRUTCH WALK

LEAN
FIG. 5B

Nominal Configuration
FIG. 5A
PUSH OFF FROM SURFACE WITH LEG A
FIG. 6A

RECONFIGURE LEG A
FIG. 6B
RETURN TO SURFACE
FIG. 7A

BEGIN LEAN MODE
WITH LEGS B & C
FIG. 7B
RECONFIGURE ALL LEGS BACK TO NOMINAL CONFIGURATION

FIG. 8B

PUSH OFF FROM SURFACE WITH LEGS B & C

FIG. 8A
RANGE OF MOTION FOR LEG
FIG. 11
SELF RIGHTING MODE

Fig. 12

Two legs tuck in while third leg pushes away from surface.

Skitter upside down.

Skitter summersaults to right itself.
SLIDER-CRANK MECHANISM
FOR JUMP MOTION
VARIEABLE DEFINITION

FIG. 13a
SLIDER-CRANK MECHANISM FOR JUMP MOTION VARIABLE DEFINITION

FIG. 13b
4-BAR LINKAGE
FOR LEANING MOTION
VARIABLE DEFINITION

FIG. 14b

ORIGINAL PAGE IS
OF POOR QUALITY
Appendix B

SKIT_3D Graphic Simulation

Figures and Program Listing
**SKIT_3D**

*Three Dimensional Graphic Simulation Program*

Computer graphic simulation is an excellent engineering tool for analyzing and designing spatial mechanisms. SKIT_3D is a three-dimensional graphic simulation program which allows the user to visualize SKITTER's spatial configurations by controlling system, leg segment, or actuator movements of a screen representation of the platform. User input is in the form of incremental positioning, direct positioning, or time-based data files which can be used for platform animation. Output is directed to a screen, plotter, or dump device.

**SYSTEM REQUIREMENTS:**

**Computer:** Hewlett-Packard 200 or 300 series computer with input knob present on the keyboard.

**Mass Storage:** One 3.5" 720 kbyte disc drive.

**Software:** Hewlett-Packard Basic 4.0 or greater with the Knob_20 bin loaded.

**Options:** Hewlett-Packard Graphics Language (HPGL) plotter.

Hewlett-Packard LaserJet printer (or equivalent).

**LOADING THE PROGRAM:**

To load the program:

1) Boot the computer into the Basic System environment.

2) Make the disc drive the default mass storage device by using the Mass Storage Is command.

3) Insert the SKIT_3D disc into the disc drive.

4) Type `LOAD "SKIT_3D"` <cr>.
5) Hit <RUN> key.
6) Program will begin.

USING THE PROGRAM:

The Program can be divided into two different sections depending on the type of data input. The Manual Mode of operation allows the user to input data directly or via the keyboard knob to control SKITTER's movements while the Data File Mode accepts time based data files for animating platform position sequences. All movements by the platform are relative to a local coordinate system on SKITTER. Figure 3 shows the orientation of the coordinate system (x-y-z) relative to the initial screen display.

MANUAL MODE

Main Menu:

The initial display shows a Z axis view of SKITTER with the main menu appearing at the bottom of the screen. The menu items can be accessed by pressing the corresponding function keys on the keyboard. Definition of the keys are as follows:

SYSTEM: Allows the user to reorient the entire platform.

PIVOT LINES: Allows the user to pivot the entire platform about a line constructed by any two feet of SKITTER.

ACTUATORS: Allows the user to reorient either a leg segment or the entire platform by engaging a particular actuator.

EXIT: Allows the user to exit the program.

OUTPUT: Allows the user to output the screen display to either a plotter or dump device.

MOVIE: Allows the user to enter DATA FILE Mode.
ATTRIBUTES: Allows the user to change views, window parameters, and output devices.

WHAT: Allows the user to view screen, output device, and platform parameters.

SYSTEM:
The SYSTEM function allows the user to rotate or translate the entire platform about or along all three axes. When the key is pressed, a new menu will appear and is defined as follows:

Rotate X: Rotates the platform incrementally about the X axis when the knob is turned.
Rotate Y: Rotates the platform incrementally about the Y axis when the knob is turned.
Rotate Z: Rotates the platform incrementally about the Z axis when the knob is turned.
Rotation Angle: Rotates the platform about the last rotation axis by a user defined angle (<cr> quits).
Knob Increment: Allows the user to input a new knob increment.
Translate X: Translates the platform incrementally along the X axis when the knob is turned.
Translate Y: Translates the platform incrementally along the y axis when the knob is turned.
Translate Z: Translates the platform incrementally along the z axis when the knob is turned.
Trans Vector: Translates the platform along a user defined vector (<cr> to quit)
Main Menu: Returns the user to the main menu.
**Pivot Lines:**

The PIVOT LINES function allows the user to pivot the entire platform about a line constructed by two of SKITTER's feet. When the key is depressed, a new menu will appear and is defined as follows:

- **Leg A:** Pivots the platform about the leg a pivot line constructed by the feet of legs b & c.
- **Leg B:** Pivots the platform about the leg b pivot line constructed by the feet of legs a & c.
- **Leg C:** Pivots the platform about the leg c pivot line constructed by the feet of legs a & b.

- **Main Menu:** Returns the user to the main menu.

**ACTUATORS:**

The ACTUATORS function key allows the user to reorient a leg segment or the entire platform by engaging a particular actuator. When the key is depressed a new menu appears and is defined as follows:

- **Femur A:** Engages the femur a actuator and will incrementally change the femur a-body angle when the knob is turned.
- **Femur B:** Engages the femur b actuator and will incrementally change the femur b-body angle when the knob is turned.
- **Femur C:** Engages the femur c actuator and will incrementally change the femur c-body angle when the knob is turned.
**Fixed/Free:** A Toggle switch which allows the user to move the platform with either its legs always in contact with the ground (fixed) or unconstrained by the ground (free).

**Knob Increment:** Allows the user to define a new knob increment.

**Tibia A:** Engages the tibia A actuator and will incrementally change the tibia-femur angle of leg a when the knob is turned.

**Tibia B:** Engages the tibia b actuator and will incrementally change the tibia-femur angle of leg b when the knob is turned.

**Tibia C:** Engages the tibia c actuator and will incrementally change the tibia-femur angle of leg c when the knob is turned.

**Main Menu:** Returns the user to the main menu.

**OUTPUT:**

The OUTPUT function key allows the user to send the screen display to a plotter or a dump device. When the key is pressed, a new menu will appear and is defined as follows:

- **Plotter:** Outputs the screen display to the designated plotter.
- **Raster Dump:** Outputs the screen display to the designated dump device.
- **Main Menu:** Returns the user to the main menu.

**ATTRIBUTES:**

The ATTRIBUTES key allows the user to define views, window parameters, and output devices. When the key is pressed, a new menu will appear and is defined as follows:

- **View:** Allows the user to define a new view. When this key is depressed, a new menu will appear and is defined as follows:
  - **X Axis:** Changes the user's view to looking down the X axis.
  - **Y Axis:** Changes the user's view to looking down the Y axis.
  - **Z Axis:** Changes the user's view to looking down the Z axis.
Quit: Returns the user to the ATTRIBUTES menu.

Window: Allows the user to change window parameters. When this key is pressed, a new menu will appear and is defined as follows:

Zoom: Allows the user to zoom in and out from the current window.

Pan X: Allows the user to pan horizontally.

Pan Y: Allows the user to pan vertically.

Input Data: Allows the user to input specific window coordinates.

Quit: Returns the user to the ATTRIBUTES menu.

Display Quantities: Displays current positions and incremental changes of the entire system or body segments as they are moved.

Dump Device: Allows the user to specify a new dump device.

Plotter Port: Allows the user to specify a new plotter port.

Main Menu: Returns the user to the main menu.

WHAT:

The WHAT function key allows the user to view the values of all parameters such as joint angles, output devices, window variables, and current view. To exit to the main menu, simply hit <cr>.

DATA FILE MODE

The MOVIE key accessible on the main menu allows the user to enter DATA FILE MODE. This mode of operation accepts time based data files created previously by the user and determines the transformation matrices for each time increment. A new file is built on the disc drive named SKITWORKS which contains the SKITTER position sequence.
After the SKITWORKS file is closed the program begins animating the position sequence on the screen.

The DATA FILE mode is extremely useful for integrating output data files from kinematic or control programs with computer graphics. The user is able to vary any particular parameter, such as mass, inertia, or gravity in his application program, create an input data file, and see graphically the effects on SKITTER as it goes through a position sequence. Once a theoretical SKITTER model is complete design of the system components can begin using the model parameters (i.e. control parameters, leg lengths, weights etc.) as design constraints.

INPUT DATA FILE STRUCTURE

The input file structure consists of values for time, free or fixed leg segment movements, and system rotations and translations. The file can be 30 lines long and each line is arranged as follows:

```
TIME FREE FEMUR FEMUR TIBIA TIBIA FEMUR TIBIA ROT ROT ROT TRAN TRAN TRAN
   A   A   B   B   C   C   X   Y   Z   X   Y   Z
```

This particular example shows that the user wishes to rotate the system around the x axis 15 degrees, move femur a 3.2 degrees in fixed mode, and tibia a 5 degrees in the fixed mode at time increment .2.

**TIME:** Time can have any value; however the value of 999 is reserved as a pointer to tell the program that it is at the end of the input data file. On the final line of the data file the user has to make the time value equal to 999.

**FREE:** If the value for free equals 0, then the foot will be fixed; however, if the value for free equals 1 then the foot is free (see manual mode - actuators menu).

When the MOVIE keys is pressed, the user will be asked:
DO YOU WANT TO RUN AN ALREADY COMPUTED FILE?  Y OR N

If the user has run the movie function before and has saved a SKITWORKS file, he may enter Y. However, if this is the first time through for the user or a brand new input file, he should enter N.

At the prompt, enter in the name of either the computed file or new input file depending on your previous answer. The program will proceed and animate the position sequence. To stop the animation sequence simply hit <cr>.

Once the animation sequence is stopped, the program will ask the user if he would like to save the SKITWORKS file as a computed file if the input data file was new. If so enter Y and enter the name of the file at the next prompt.

A sample input file, SHOW, and a computed file, WALK, are stored on the SKIT_3D disc and can be used to demonstrate the DATA FILE mode for the user.
THESE ARE THE CURRENT PARAMETERS:

WINDOW:
X MIN = -20
Y MIN = -20
X MAX = 40
Y MAX = 40

VIEW: LOOKING DOWN Z AXIS

PLOTTER LOCATION: 705

DISPLAY VALUES IS OFF

<table>
<thead>
<tr>
<th>TRANS (in.)</th>
<th>ROT (deg)</th>
<th>FEMUR ANGLE</th>
<th>TIBIA ANGLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Y</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Z</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

PIVOT ANG A: 0  PIVOT ANG B: 0  PIVOT ANG C: 0

FORM FEED   | LASER PRINTER | SCREEN   | HARD DISK   | DISK DRIVE
SCRATCH     | LOAD ""       | CATALOG DRIVE | LIST PROGRAM | RE-STORE ""

TYPICAL OUTPUT FROM 'WHAT' COMMAND
**SKIT 3D V.10**

**THREE DIMENSIONAL GRAPHIC SIMULATION PROGRAM**

**WRITTEN BY:**

BRICE K. MACLAREN

GARY V. MCMURRAY

06/23/88

**HARDWARE:** HEWLETT PACKARD 200/300 SERIES COMPUTER

3.5' DISC DRIVE

KEYBOARD WITH KNOB

**SOFTWARE:** HEWLETT-PACKARD BASIC 4.0 WITH KNOB_20 BIN LOADED

**OPTIONS:** HEWLET-PACKARD (HPGL) PLOTTER

**PRINTER**

---

**SKITTER DATA FILE:** SEE MANUAL FOR DESCRIPTION OF BODY POINTS

---

**DATA**

**UPPER BODY**

<table>
<thead>
<tr>
<th>DATA</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.5482,28.4912,0,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,28.4912,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,28.4912,5.6709,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-7.3759,20.9817,6.3188,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-7.3759,20.9817,-6.3188,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,28.4912,-5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,28.4912,-5.6709,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,-9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,-3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.5482,28.4912,0,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,28.4912,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.5482,13.4722,0,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.5482,13.4722,5.6709,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BOTTOM BODY**

<table>
<thead>
<tr>
<th>DATA</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.5482,13.4722,0,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.5482,13.4722,5.6709,-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1602,20.9817,3.2283,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1.7468,20.9817,9.5688,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3.2741,13.4722,5.6709,-1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**NOTE:**

The data file includes the coordinates for the body points of the Skitter mobile platform. Each line represents a point in 3D space, with the first three numbers indicating the X, Y, Z coordinates, and the last number indicating the angle in degrees. The program uses these points to accurately depict the platform and its motions when inputted through manual or data file input. Use function keys and knob for input.

**HARDWARE:** HEWLETT PACKARD 200/300 SERIES COMPUTER

3.5' DISC DRIVE

KEYBOARD WITH KNOB

**SOFTWARE:** HEWLETT-PACKARD BASIC 4.0 WITH KNOB_20 BIN LOADED

**OPTIONS:** HEWLET-PACKARD (HPGL) PLOTTER

**PRINTER**
<table>
<thead>
<tr>
<th>DATA</th>
<th>-3.2741, 13.4722, -5.6709, -1</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>520</td>
<td>-3.2741, 13.4722, -5.6709, -2</td>
<td>F</td>
</tr>
<tr>
<td>530</td>
<td>-1.7468, 20.9817, -9.5688, -1</td>
<td>L</td>
</tr>
<tr>
<td>540</td>
<td>9.1602, 20.9817, -3.2283, -1</td>
<td>G</td>
</tr>
<tr>
<td>550</td>
<td>6.5482, 13.4722, 0, -1</td>
<td>D</td>
</tr>
<tr>
<td>560</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>580</td>
<td>0, 0, 0, 4</td>
<td></td>
</tr>
<tr>
<td>590</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>600</td>
<td>9.1602, 20.9817, -3.2283, -2</td>
<td>G</td>
</tr>
<tr>
<td>610</td>
<td>29.6602, 20.9817, -3.2283, -1</td>
<td>M</td>
</tr>
<tr>
<td>620</td>
<td>18.7362, 25.0731, 0, -1</td>
<td>O</td>
</tr>
<tr>
<td>630</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>640</td>
<td></td>
<td></td>
</tr>
<tr>
<td>650</td>
<td></td>
<td></td>
</tr>
<tr>
<td>660</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>670</td>
<td>29.6602, 20.9817, 3.2283, -2</td>
<td>N</td>
</tr>
<tr>
<td>680</td>
<td>18.7362, 25.0731, 0, -1</td>
<td>O</td>
</tr>
<tr>
<td>690</td>
<td>9.1602, 20.9817, 3.2283, -1</td>
<td>H</td>
</tr>
<tr>
<td>700</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>710</td>
<td></td>
<td></td>
</tr>
<tr>
<td>720</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>730</td>
<td>9.1602, 20.9817, -3.2283, -2</td>
<td>G</td>
</tr>
<tr>
<td>740</td>
<td>29.6602, 20.9817, -3.2283, -1</td>
<td>M</td>
</tr>
<tr>
<td>750</td>
<td>20.571, 16.8903, 0, -1</td>
<td>P</td>
</tr>
<tr>
<td>760</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>770</td>
<td></td>
<td></td>
</tr>
<tr>
<td>780</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>790</td>
<td>29.6602, 20.9817, 3.2283, -2</td>
<td>N</td>
</tr>
<tr>
<td>800</td>
<td>20.571, 16.8903, 0, -1</td>
<td>P</td>
</tr>
<tr>
<td>810</td>
<td>9.1602, 20.9817, 3.2283, -1</td>
<td>H</td>
</tr>
<tr>
<td>820</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>830</td>
<td></td>
<td></td>
</tr>
<tr>
<td>840</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>850</td>
<td>29.6602, 20.9817, 3.2283, -2</td>
<td>M</td>
</tr>
<tr>
<td>860</td>
<td>30.4255, 0, 0, -1</td>
<td>A'</td>
</tr>
<tr>
<td>870</td>
<td>35.7901, 9.2132, 0, -1</td>
<td>B'</td>
</tr>
<tr>
<td>880</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>890</td>
<td></td>
<td></td>
</tr>
<tr>
<td>900</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>910</td>
<td>29.6602, 20.9817, 3.2283, -2</td>
<td>N</td>
</tr>
<tr>
<td>920</td>
<td>30.4255, 0, 0, -1</td>
<td>A'</td>
</tr>
<tr>
<td>930</td>
<td>35.7901, 9.2132, 0, -1</td>
<td>B'</td>
</tr>
<tr>
<td>940</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>950</td>
<td></td>
<td></td>
</tr>
<tr>
<td>960</td>
<td>0, 0, 0, 4</td>
<td></td>
</tr>
<tr>
<td>970</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>980</td>
<td>-1.7468, 20.9817, 9.5688, -2</td>
<td>I</td>
</tr>
<tr>
<td>990</td>
<td>-9.3493, 25.0731, 16.2368, -1</td>
<td>S</td>
</tr>
<tr>
<td>1000</td>
<td>-11.9968, 20.9817, 27.3223, -1</td>
<td>Q</td>
</tr>
<tr>
<td>1010</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
<tr>
<td>1020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1030</td>
<td>0, 0, 0, 10</td>
<td></td>
</tr>
<tr>
<td>1040</td>
<td>-17.6259, 20.9817, 24.0723, -2</td>
<td>R</td>
</tr>
<tr>
<td>1050</td>
<td>-9.3493, 25.0731, 16.2368, -1</td>
<td>S</td>
</tr>
<tr>
<td>1060</td>
<td>-7.3759, 20.9817, 6.3188, -1</td>
<td>J</td>
</tr>
<tr>
<td>1070</td>
<td>0, 0, 0, 7</td>
<td></td>
</tr>
</tbody>
</table>

**FEMUR ONE**

**TIBIA ONE**

**FEMUR TWO**

*RESERVED FOR PEN*
<table>
<thead>
<tr>
<th>R080</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1090</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1100</td>
<td>DATA</td>
<td>-1.7468,20.9817,9.5688,-2</td>
<td></td>
</tr>
<tr>
<td>1110</td>
<td>DATA</td>
<td>-10.2667,16.8903,17.8258,-1</td>
<td></td>
</tr>
<tr>
<td>1120</td>
<td>DATA</td>
<td>-11.9968,20.9817,27.3223,-1</td>
<td></td>
</tr>
<tr>
<td>1130</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1140</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1150</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1160</td>
<td>DATA</td>
<td>-17.6259,20.9817,24.0723,-2</td>
<td></td>
</tr>
<tr>
<td>1170</td>
<td>DATA</td>
<td>-10.2667,16.8903,17.8258,-1</td>
<td></td>
</tr>
<tr>
<td>1180</td>
<td>DATA</td>
<td>-7.3759,20.9817,6.3188,-1</td>
<td></td>
</tr>
<tr>
<td>1190</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1200</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1210</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1220</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1230</td>
<td>DATA</td>
<td>-11.9968,20.9817,27.3223,-2</td>
<td></td>
</tr>
<tr>
<td>1240</td>
<td>DATA</td>
<td>-15.1940,0,26.3601,-1</td>
<td></td>
</tr>
<tr>
<td>1250</td>
<td>DATA</td>
<td>-17.8763,9.2132,31.0059,-1</td>
<td></td>
</tr>
<tr>
<td>1260</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1270</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1280</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1290</td>
<td>DATA</td>
<td>-17.6259,20.9817,24.0723,-2</td>
<td></td>
</tr>
<tr>
<td>1300</td>
<td>DATA</td>
<td>-15.1940,0,26.3601,-1</td>
<td></td>
</tr>
<tr>
<td>1310</td>
<td>DATA</td>
<td>-17.8763,9.2132,31.0059,-1</td>
<td></td>
</tr>
<tr>
<td>1320</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1330</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1340</td>
<td>DATA</td>
<td>0,0,0,4</td>
<td></td>
</tr>
<tr>
<td>1350</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1360</td>
<td>DATA</td>
<td>-7.3759,20.9817,-6.3188,-2</td>
<td></td>
</tr>
<tr>
<td>1370</td>
<td>DATA</td>
<td>-9.3493,25.0731,-16.2368,-1</td>
<td></td>
</tr>
<tr>
<td>1380</td>
<td>DATA</td>
<td>-17.6259,20.9817,-24.0723,-1</td>
<td></td>
</tr>
<tr>
<td>1390</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1410</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1420</td>
<td>DATA</td>
<td>-11.9968,20.9817,-27.3223,-2</td>
<td></td>
</tr>
<tr>
<td>1430</td>
<td>DATA</td>
<td>-9.3493,25.0731,-16.2368,-1</td>
<td></td>
</tr>
<tr>
<td>1440</td>
<td>DATA</td>
<td>-1.7468,20.9817,-9.5688,-1</td>
<td></td>
</tr>
<tr>
<td>1450</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1460</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1470</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1480</td>
<td>DATA</td>
<td>-7.3759,20.9817,-6.3188,-2</td>
<td></td>
</tr>
<tr>
<td>1490</td>
<td>DATA</td>
<td>-10.2667,16.8903,-17.8258,-1</td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>DATA</td>
<td>-17.6259,20.9817,-24.0723,-1</td>
<td></td>
</tr>
<tr>
<td>1510</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1520</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1530</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1540</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1550</td>
<td>DATA</td>
<td>-11.9968,20.9817,-27.3223,-2</td>
<td></td>
</tr>
<tr>
<td>1560</td>
<td>DATA</td>
<td>-10.2667,16.8903,-17.8258,-1</td>
<td></td>
</tr>
<tr>
<td>1570</td>
<td>DATA</td>
<td>-1.7468,20.9817,-9.5688,-1</td>
<td></td>
</tr>
<tr>
<td>1580</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1590</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1600</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1610</td>
<td>DATA</td>
<td>-17.6259,20.9817,-24.0723,-2</td>
<td></td>
</tr>
<tr>
<td>1620</td>
<td>DATA</td>
<td>-15.1940,0,-26.3601,-1</td>
<td></td>
</tr>
<tr>
<td>1630</td>
<td>DATA</td>
<td>-17.8763,9.2132,-31.0059,-1</td>
<td></td>
</tr>
<tr>
<td>1640</td>
<td>DATA</td>
<td>0,0,0,7</td>
<td></td>
</tr>
<tr>
<td>1650</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1660</td>
<td>DATA</td>
<td>0,0,0,10</td>
<td></td>
</tr>
<tr>
<td>1670</td>
<td>DATA</td>
<td>-11.9968,20.9817,-27.3223,-2</td>
<td></td>
</tr>
</tbody>
</table>

**TIBIA TWO**

**FEMUR THREE**

**TIBIA THREE**
DATA -15.1940,0,-26.3601,-1  ! E'  
DATA -17.8763,9.2132,-31.0059,-1 ! F'  
DATA 0,0,0,7  
REM  
******************************************************************  
REM MAIN PROGRAM  
REM  
******************************************************************  
OPTION BASE 1  
REAL Skitter(129,4),Newskit(129,3)  ! DEFINE VAR  
REAL Trans(4,4),Temp(129,4),Tempa(129,4)  ! TRANSFORMATION MATRIX  
REAL Total(4,4),Skitmod(129,4)  ! TOTAL TRANSFORM MATRIX  
REAL Femur(31,4),Femurmod(31,4),Femurtemp(31,4)  
REAL Tibia(10,4),Tibiastemp(10,4),Tibiamod(10,4)  
GOSUB Init  ! INITIALIZATION ROUTINE  
CALL Display_skit(Skitter(*),Newskit(*),Screen_x,Screen_y)  ! DRAW SKITTER  
Menu:  
SELECT Menu$  
CASE "MAIN"  
ON KEY 0 LABEL "SYSTEM" GOTO System  
ON KEY 2 LABEL "ACT'ATORS" GOTO Actuator  
ON KEY 5 LABEL "OUTPUT" GOTO Output  
ON KEY 7 LABEL "ATTRIBUTES" GOTO Attributes  
ON KEY 9 LABEL "WHAT ??" GOTO What  
ON KEY 4 LABEL "EXIT" GOTO Finished  
ON KEY 1 LABEL "PIVOT LINES" GOTO Pivotlines  
ON KEY 6 LABEL "MOVIE" GOTO Movie  
ON KEY 3 LABEL "" GOTO Main  
ON KEY 8 LABEL "" GOTO Main  
GOTO 1970  
CASE "PIVOTLINES"  
ON KNOB .55 GOTO Knob_isr  
ON KEY 0 LABEL "LEG A" GOSUB Pivotlega  
ON KEY 1 LABEL "" GOTO Pivotlines  
ON KEY 2 LABEL "LEG B" GOSUB Pivotlegb  
ON KEY 3 LABEL "" GOTO Pivotlines  
ON KEY 4 LABEL "LEG C" GOSUB Pivotlegc  
ON KEY 5 LABEL "" GOTO Pivotlines  
ON KEY 6 LABEL "" GOTO Pivotlines  
ON KEY 7 LABEL "" GOTO Pivotlines  
ON KEY 8 LABEL "" GOTO Pivotlines  
ON KEY 9 LABEL "MAIN MENU" GOTO Main  
GOTO 2100  
CASE "SYSTEM"  
ON KNOB .2 GOTO Knob_isr  
ON KEY 1 LABEL "ROTATE Y" GOSUB Rot_y  
ON KEY 0 LABEL "ROTATE X" GOSUB Rot_x  
ON KEY 2 LABEL "ROTATE Z" GOSUB Rot_z  
ON KEY 5 LABEL "TRANSLATE X" GOSUB Trans_x  
ON KEY 6 LABEL "TRANSLATE Y" GOSUB Trans_y
ON KEY 7 LABEL "TRANSLATE Z" GOSUB Trans_z
ON KEY 9 LABEL "MAIN MENU" GOTO Main
ON KEY 8 LABEL "TRANS VECTOR" GOSUB Vector
ON KEY 3 LABEL "ROTATION ANGLE" GOSUB Angle
ON KEY 4 LABEL "KNB INCREMENT" GOTO Increment
GOTO 2230

CASE "ATTRIBUTES"
ON KEY 0 LABEL "VIEWS" GOTO Windowpane
ON KEY 9 LABEL "MAIN" GOTO Main
ON KEY 1 LABEL "" GOTO Attributes
ON KEY 2 LABEL "WINDOW" GOTO Windows
ON KEY 3 LABEL "" GOTO Attributes
ON KEY 4 LABEL "DISP QUANTITY" GOTO Print_flag
ON KEY 5 LABEL "DUMP DEVICE" GOTO Printer
ON KEY 6 LABEL "PLOTTER PORT" GOTO Plotter
ON KEY 7 LABEL "" GOTO Attributes
ON KEY 8 LABEL "" GOTO Attributes
GOTO 2360

CASE "OUTPUT"
ON KEY 0 LABEL "PLOT" GOTO Plot
ON KEY 1 LABEL "" GOTO Output
ON KEY 2 LABEL "RASTER DUMP" GOTO Dump
ON KEY 3 LABEL "" GOTO Output
ON KEY 4 LABEL "" GOTO Output
ON KEY 5 LABEL "" GOTO Output
ON KEY 6 LABEL "" GOTO Output
ON KEY 7 LABEL "" GOTO Output
ON KEY 8 LABEL "" GOTO Output
ON KEY 9 LABEL "MAIN MENU" GOTO Main
GOTO 2500

CASE "ACTUATOR"
ON KNOB .21 GOTO Knob_leg_isr
ON KEY 0 LABEL "FEMUR A" GOTO Femur_a
ON KEY 1 LABEL "FEMUR B" GOTO Femur_b
ON KEY 2 LABEL "FEMUR C" GOTO Femur_c
ON KEY 3 LABEL "TIBIA A" GOTO Tibia_a
ON KEY 4 LABEL "TIBIA B" GOTO Tibia_b
ON KEY 5 LABEL "TIBIA C" GOTO Tibia_c
ON KEY 6 LABEL "" GOTO Menu
ON KEY 3 LABEL Freeleg$ GOTO Free_leg
ON KEY 4 LABEL "KNB INCREMENT" GOTO Increment
ON KEY 9 LABEL "MAIN MENU" GOTO Main
GOTO 2640

END SELECT

REM**************************************************************
REM KNOB ISR: KNOB INTERRUPT SERVICE ROUTINE. ON KNOB ROTATION, THE
REM APPROPRIATE FUNCTION WILL BE CARRIED OUT BY THE KNOB INCREMENT
REM AMOUNT
REM**************************************************************

Knob_isr:
Theta=Increment*SGN(KNOBX)
SELECT Twirl$
ITEM SYSTEM ROTATION

2702 CASE "ROTATE Y"
2710 Theta=Theta
2720 CALL Rotate_y(Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Sys_rot_y,Theta,Printflag$)
2730 !
2740 CASE "ROTATE X"
2750 CALL Rotate_x(Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Sys_rot_x,Theta,Printflag$)
2760 !
2770 CASE "ROTATE Z"
2780 CALL Rotate_z(Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Sys_rot_z,Theta,Printflag$)
2790 !
2791 REM SYSTEM TRANSLATION
2800 CASE "TRANSLATE"
2810 CALL Translate_3d(Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Sys_trans_x,Sys_trans_y,Sys_trans_z,Way,Theta,Printflag$)
2820 !
2821 REM PIVOT LINES
2830 CASE "PIVOTLINES"
2840 Theta=Theta
2850 
2860 IF Leg_flag$="FEMURA" THEN
2870 Pivot_ang_a=Pivot_ang_a+Theta
2880 DISP "MODE: ROTATE ABOUT PIVOT LINE A":Theta;
2890 Pivotx=(Skitter(96,1)+Skitter(127,1))/2
2900 Pivoty=(Skitter(96,2)+Skitter(127,2))/2
2910 Pivotz=(Skitter(96,3)+Skitter(127,3))/2
2920 Dist_piv_foot=SQR((Pivotx-Skitter(65,1))^2+(Pivoty-Skitter(65,2))^2+(Pivotz-Skitter(65,3))^2)
2930 CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,-Pivotz)
2940 Flag=0
2950 CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,-Theta,Pivotx,Pivoty,Pivotz)
2960 CALL Trans_fem_orig(Skitmod(*),Skitmod(*),Pivotx,Pivoty,Pivotz)
2970 END IF

2980 IF Leg_flag$="FEMUR B" THEN
2990 Pivot_ang_b=Pivot_ang_b+Theta
3000 DISP "MODE: ROTATE ABOUT PIVOT LINE B BY":Theta;
3010 Pivotx=(Skitter(65,1)+Skitter(127,1))/2
3020 Pivoty=(Skitter(65,2)+Skitter(127,2))/2
3030 Pivotz=(Skitter(65,3)+Skitter(127,3))/2
3040 Dist_piv_foot=SQR((Pivotx-Skitter(96,1))^2+(Pivoty-Skitter(96,2))^2+(Pivotz-Skitter(96,3))^2)
3050 CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,-Pivotz)
3060 CALL Rotate_leg_y(Skitmod(*),Skitmod(*),60)
80 Flag=0
90 CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Theta,Pr

CALL Rotate_leg_y(Skitmod(*),Skitmod(*),-60)
CALL Trans_fem_orig(Skitmod(*),Skitmod(*),Pivotx,Pivoty,Pivotz)
MAT Skitter= Skitmod
END IF

IF Leg_flag$="FEMUR C" THEN
Pivot_ang_c=Pivot_ang_c+Theta
DISP " MODE: ROTATE ABOUT PIVOT LINE C BY:";Theta;

Pivotx=(Skitter(65,1)+Skitter(96,1))/2
Pivoty=(Skitter(65,2)+Skitter(96,2))/2
Pivotz=(Skitter(65,3)+Skitter(96,3))/2
Dist_pivot_foot=SQR((Pivotx-Skitter(127,1))^2+(Pivoty-Skitter(127,2))^2+(Pivotz-Skitter(127,3))^2)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,-Pivotz)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),-60)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Theta,Pr
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),60)
CALL Transfemorlg(Skitmod(*),Skitmod(*),Pivotx,Pivoty,-Pivotz)

CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Theta,Pr
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),60)
CALL Transfemorlg(Skitmod(*),Skitmod(*),Pivotx,Pivoty,-Pivotz)

MAT Skitter= Skitmod
END IF

MAT Tempa(*,1:3)= Skitter(*,1:3)
MAT Temp= Tempa*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)
END SELECT
CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y)
GOTO Menu

REM%%%%%%%%%%%%%%%%%%%%%% REM KNOB LEG_ISR: KNOB LEG INTERRUPT SERVICE ROUTINE FOR ACTUATOR MENU
REM%%%%%%%%%%%%%%%%%%%%%%

Knob_leg_isr:
Theta=Increment*SGN(KNOBX)
SELECT Legs
CASE "FEMUR A"
  Theta=Theta
  MAT Femur= Skitter(37:67,*)
  CALL Trans_fem_orig(Femur(*),Femurmod(*),-Femur(3,1),
  Femur(3,2),-Femur(3,3))
  Flag=1
  CALL Rotate_leg_y(Femurmod(*),Femurmod(*),180)
  CALL Rotate_leg_z(Femurmod(*),Femurmod(*),Leg$,Theta,
  intflag$,Fem_a_ang,Flag)
  CALL Rotate_leg_y(Femurmod(*),Femurmod(*),180)
CALL Trans_fem_orig(Femurmod(*),Femurmod(*),Femur(3,1)
MAT Skitter(37:67,*)= Femurmod

IF Freeleg$="FREE" THEN
MAT Femurtemp(*,1:3)= Femurmod(*,1:3)
MAT Femur= Femurtemp*Total
MAT Skitmod(37:67,1:3)= Femur(*,1:3)
GOTO End_leg
END IF

Pivotx=(Skitter(96,1)+Skitter(127,1))/2
Pivory=(Skitter(96,2)+Skitter(127,2))/2
Pivoz=(Skitter(96,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(65,1))^2+(Pivory-Skitter(65,2))^2+(Pivoz-Skitter(65,3))^2)
Dist_y=Pivory-Skitter(65,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivory,-Pivoz)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,,-Rot_ang

MAT Skitter= Skitmod
MAT Temp(*,1:3)= Skitter(*,1:3)
MAT Temp= Tempa*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)

CASE "FEMUR B"
MAT Femur= Skitter(68:98,*)
CALL Trans_fem_orig(Femur(*),Femurmod(*),-Femur(3,1),
Flag=1
CALL Rotate_leg_y(Femurmod(*),Femurmod(*),60)
CALL Rotate_leg_z(Femurmod(*),Femurmod(*),Leg$,,-Theta,
CALL Rotate_leg_y(Femurmod(*),Femurmod(*),-60)
CALL Trans_fem_orig(Femurmod(*),Femurmod(*),Femur(3,1
MAT Skitter(68:98,*)= Femurmod

IF Freeleg$="FREE" THEN
MAT Femurtemp(*,1:3)= Femurmod(*,1:3)
MAT Femur= Femurtemp*Total
MAT Skitmod(68:98,1:3)= Femur(*,1:3)
GOTO End_leg
END IF

Pivotx=(Skitter(65,1)+Skitter(127,1))/2
Pivory=(Skitter(65,2)+Skitter(127,2))/2
Pivotz=(Skitter(65,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(96,1))^2+(Pivory-Skitter(96,2))^2+(Pivotz-Skitter(96,3))^2)
Dist_y=Pivory-Skitter(96,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivory,-Pivotz)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),60)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Rot_ang,
rintflag$,Fem_b_ang,Flag)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),-60)
CALL Trans_fem_orig(Skitmod(*),Skitmod(*),Pivotx,Pivoty,
Pivotz)
MAT Skitter= Skitmod
MAT Temp(*,1:3)= Skitter(*,1:3)
MAT Temp= Temp*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)
CASE "FEMUR C"
Theta=-Theta
MAT Femur= Skitter(99:129,*)
CALL Trans_fem_orig(Femur(*),Femurmod(*),-Femur(3,1),
Femur(3,2),-Femur(3,3))
CALL Rotate_leg_y(Femurmod(*),Femurmod(*),-60)
Flag=1
CALL Rotate_leg_z(Femurmod(*),Femurmod(*),Leg$,Theta,
,Femur(3,2),Femur(3,3))
CALL Rotate_leg_y(Femurmod(*),Femurmod(*),60)
CALL Trans_fem_orig(Femurmod(*),Femurmod(*),Femur(3,1)
MAT Skitter(99:129,*)= Femurmod
IF Freeleg$="FREE" THEN
MAT Femurtemp(*,1:3)= Femurmod(*,1:3)
MAT Femur= Femurtemp*Total
MAT Skitmod(99:129,1:3)= Femur(*,1:3)
GOTO End_leg
END IF
Pivotx=(Skitter(65,1)+Skitter(96,1))/2
Pivoty=(Skitter(65,2)+Skitter(96,2))/2
Pivotz=(Skitter(65,3)+Skitter(96,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(127,1))^2+(Pivoty-Skitter(127,2))^2)
Dist_y=Pivoty-Skitter(127,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,
,Pivotz)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),-60)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Rot_ang,
rintflag$,Fem_c_ang,Flag)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),60)
CALL Trans_fem_orig(Skitmod(*),Skitmod(*),Pivotx,Pivoty,
Pivotz)
MAT Skitter= Skitmod
MAT Temp(*,1:3)= Skitter(*,1:3)
MAT Temp= Temp*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)
CASE "TIBIA A"
Theta=-Theta
MAT Tibia= Skitter(58:67,*)
CALL Trans_fem_orig(Tibia(*),Tibiamod(*),-Tibia(2,1),
Flag=1
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),180)
CALL Rotate_leg_z(Tibiamod(*),Tibiamod(*),Leg$,Theta,
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),180)
CALL Trans_fem_orig(Tibiamod(*),Tibiamod(*),Tibia(2,1)
MAT Skitter(58:67,*)= Tibiamod

IF Freeleg$="FREE" THEN
MAT Tibiatemp(*,1:3)= Tibiamod(*,1:3)
MAT Tibia= Tibiatemp*Total
MAT Skitmod(58:67,1:3)= Tibia(*,1:3)
GOTO End_leg
END IF
END IF

Pivotx=(Skitter(96,1)+Skitter(127,1))/2
Pivoty=(Skitter(96,2)+Skitter(127,2))/2
Pivots=(Skitter(96,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(65,1))^2+(Pivoty-Sk
Dist_y=Pivoty-Skitter(65,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivo
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,,-Rot_ang
CALL Trans_fem_orig(Skitmod(*),Skitmod(*),Pivotx,Pivo
MAT Skitter= Skitmod
MAT Temp(*,1:3)= Skitter(*,1:3)
MAT Temp= Temp*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)

CASE "TIBIA B"
MAT Tibia= Skitter(89:98,*)
CALL Trans_fem_orig(Tibia(*),Tibiamod(*),-Tibia(2,1),
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),60)
Flag=1
CALL Rotate_leg_z(Tibiamod(*),Tibiamod(*),Leg$,Theta,
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),-60)
CALL Trans_fem_orig(Tibiamod(*),Tibiamod(*),Tibia(2,1
MAT Skitter(89:98,*)= Tibiamod

IF Freeleg$="FREE" THEN
MAT Tibiatemp(*,1:3)= Tibiamod(*,1:3)
MAT Tibia= Tibiatemp*Total
MAT Skitmod(89:98,1:3)= Tibia(*,1:3)
GOTO End_leg
END IF

Pivotx=(Skitter(65,1)+Skitter(127,1))/2
Pivoty\(=(\text{Skitter}(65,2)+\text{Skitter}(127,2))/2\)

Pivotz\(=(\text{Skitter}(65,3)+\text{Skitter}(127,3))/2\)

\[
\text{Dist}_{\text{piv}}_{\text{foot}}=\sqrt{(\text{Pivotx}-\text{Skitter}(127,1))^2+(\text{Pivoty}-\text{Skitter}(127,2))^2+(\text{Pivotz}-\text{Skitter}(127,3))^2}
\]

\[
\text{Dist}_y=\text{Pivoty}-\text{Skitter}(127,2)
\]

\[
\text{Rot}_\text{ang}=	ext{ASN}(\text{Dist}_y/\text{Dist}_{\text{piv}}_{\text{foot}})
\]

CALL Trans\_fem\_orig\(\text{Skitter}(\cdot),\text{Skitmod}(\cdot),-\text{Pivotx},-\text{Pivoty},-\text{Pivotz}\)

CALL Rotate\_leg\_y\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),60\)

Flag=0

CALL Rotate\_leg\_z\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),\text{Leg}_{\cdot}\),\text{Rot}_\text{ang},

CALL Rotate\_leg\_y\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),-60\)

CALL Trans\_fem\_orig\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),\text{Pivotx},\text{Pivoty},\text{Pivotz}\)

MAT \text{Skitter}=\text{Skitmod}

MAT \text{Tempa}(\cdot,1:3)=\text{Skitter}(\cdot,1:3)

MAT \text{Temp}=\text{Tempa}\cdot\text{Total}

MAT \text{Skitmod}(\cdot,1:3)=\text{Temp}(\cdot,1:3)

CASE "TIBIA C"

MAT \text{Tibia}=\text{Skitter}(120:129,\cdot)

CALL Trans\_fem\_orig\(\text{Tibia}(\cdot),\text{Tibiamod}(\cdot),-\text{Tibia}(2,1),\)

CALL Rotate\_leg\_y\(\text{Tibiamod}(\cdot),\text{Tibiamod}(\cdot),-60\)

Flag=1

CALL Rotate\_leg\_z\(\text{Tibiamod}(\cdot),\text{Tibiamod}(\cdot),\text{Leg}_{\cdot}\),\text{Theta},

CALL Rotate\_leg\_y\(\text{Tibiamod}(\cdot),\text{Tibiamod}(\cdot),60\)

CALL Trans\_fem\_orig\(\text{Tibiamod}(\cdot),\text{Tibiamod}(\cdot),\text{Tibia}(2,1)

MAT \text{Skitter}(120:129,\cdot)=\text{Tibiamod}

IF Freeleg$="FREE" THEN

MAT \text{Tibiatemp}(\cdot,1:3)=\text{Tibiamod}(\cdot,1:3)

MAT \text{Tibia}=\text{Tibiatemp}\cdot\text{Total}

MAT \text{Skitmod}(120:129,1:3)=\text{Tibia}(\cdot,1:3)

GOTO End\_leg

END IF

\[
\text{Pivotx}=(\text{Skitter}(65,1)+\text{Skitter}(96,1))/2
\]

\[
\text{Pivoty}=(\text{Skitter}(65,2)+\text{Skitter}(96,2))/2
\]

\[
\text{Pivotz}=(\text{Skitter}(65,3)+\text{Skitter}(96,3))/2
\]

\[
\text{Dist}_{\text{piv}}_{\text{foot}}=\sqrt{(\text{Pivotx}-\text{Skitter}(127,1))^2+(\text{Pivoty}-\text{Skitter}(127,2))^2+(\text{Pivotz}-\text{Skitter}(127,3))^2}
\]

\[
\text{Dist}_y=\text{Pivoty}-\text{Skitter}(127,2)
\]

\[
\text{Rot}_\text{ang}=	ext{ASN}(\text{Dist}_y/\text{Dist}_{\text{piv}}_{\text{foot}})
\]

CALL Trans\_fem\_orig\(\text{Skitter}(\cdot),\text{Skitmod}(\cdot),-\text{Pivotx},-\text{Pivoty},-\text{Pivotz}\)

CALL Rotate\_leg\_y\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),60\)

Flag=0

CALL Rotate\_leg\_z\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),\text{Leg}_{\cdot}\),\text{Rot}_\text{ang},

CALL Rotate\_leg\_y\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),-60\)

CALL Trans\_fem\_orig\(\text{Skitmod}(\cdot),\text{Skitmod}(\cdot),\text{Pivotx},\text{Pivoty},\text{Pivotz}\)

MAT \text{Skitter}=\text{Skitmod}

MAT \text{Tempa}(\cdot,1:3)=\text{Skitter}(\cdot,1:3)

MAT \text{Temp}=\text{Tempa}\cdot\text{Total}
MAT Skitmod(*,1:3)= Temp(*,1:3)

5500  !
5510  End_leg:  !
5520      END SELECT
5530  !
5540      CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y)
5550  GOTO Menu
5560  REM ***************************************************************
5570  REM GOSUB ROUTINES FOR MENU AND KNOB ISR CASE SELECTION
5580  REM ***************************************************************
5590  Main:  !
5590  Menu$="MAIN"
5600  GOTO Menu
5600  Movie:  !
5610      CALL Movie
5620      RESTORE 160
5630      CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y)
5640  GOTO Menu
5650  Pivotlines:  !
5660      Menu$="PIVOTLINES"
5670  GOTO Menu
5680  Pivotlega:  !
5690      Twirl$="PIVOTLINES"
5700      Leg_flag$="FEMUR A"
5710  RETURN
5730  !
5740  Pivotlegb:  !
5750      Twirl$="PIVOTLINES"
5760      Leg_flag$="FEMUR B"
5770  RETURN
5780  !
5790  Pivotlegc:  !
5800      Twirl$="PIVOTLINES"
5810      Leg_flag$="FEMUR C"
5820  RETURN
5830  !
5840  System:  !
5850      Menu$="SYSTEM"
5860  GOTO Menu
5870  !
5880  Attributes:  !
5890      Menu$="ATTRIBUTES"
5900  GOTO Menu
5910  !
5920  Attributes:  !
5930      Menu$="ATTRIBUTES"
5940  GOTO Menu
5950  !
5960  !
5970  Actuator:  !
5980  Menu$="ACTUATOR"
5990  GOTO Menu
6000  !
6010  Increment:  !
6020  DISP " INPUT NEW INCREMENT CURRENT VALUE: ",Increment;
6030  LINPUT Increment$
5040 IF Increment$="" THEN
5050 GOTO 6090
5060 ELSE
5070 Increment=VAL(Increment$)
5080 END IF
5090 GOTO Menu
5100 Free_leg:
5110 IF Freeleg$="FIXED" THEN
5120 DISP " LEG IS NOW FREE TO ROTATE"
5130 Freeleg$="FREE"
5140 ELSE
5150 DISP " LEG IS NOW FIXED "
5160 Freeleg$="FIXED"
5170 END IF
5180 GOTO Menu
5190 Femur_a:
5200 Leg$="FEMUR A"
5210 DISP " MODE: FEMUR A"
5220 GOTO Menu
5230 1
5240 Femur_b:
5250 Leg$="FEMUR B"
5260 DISP " MODE: FEMUR B"
5270 GOTO Menu
5280 1
5290 Femur_c:
5300 Leg$="FEMUR C"
5310 DISP " MODE: FEMUR C"
5320 GOTO Menu
5330 1
5340 Tibia_a:
5350 Leg$="TIBIA A"
5360 DISP " MODE: TIBIA A"
5370 GOTO Menu
5380 1
5390 Tibia_b:
5400 Leg$="TIBIA B"
5410 DISP " MODE: TIBIA B"
5420 GOTO Menu
5430 1
5440 Tibia_c:
5450 Leg$="TIBIA C"
5460 DISP " MODE: TIBIA C"
5470 GOTO Menu
5480 1
5490 Windows:
5500 CALL zoom_pan(Window$,Screenx_win_min,Screenx_win_max,Screeny_win_min,Screeny_win_max,Skimod(*),Neukskit(*),Screen_x,Screen_y)
5510 GOTO Menu
5520 What:
5530 1
5540 Ap=Screenx_win_min
5550 Bp=Screenx_win_max
5560 Cp=Screeny_win_min
5570 Dp=Screeny_win_max
5580 Ep=Sys_trans_x
5590 Fp=Sys_trans_y
5600 Gp=Sys_trans_z
5610 Hp=Sys_rot_x
5620 Ip=Sys_rot_y
GOTO Menu

CALL Windows(Skitmod(*),Newskit(*),Screen_x,Screen_y)
GOTO Menu

CALL Windows_limits(Skitmod(*),Newskit(*),Screen_x,Screen_y,Screenx_win_min,Screenx_win_max,Screeny_win_min,Screeny_win_max)
GOTO Menu

IF Prtflag$="OFF" THEN
   DISP " DISPLAY QUANTITIES IS ";CHR$(129);" ON ";CHR$(128)
   Prtflag$="ON"
ELSE
   DISP " DISPLAY QUANTITIES IS OFF"
   Prtflag$="OFF"
END IF
GOTO Menu

Twirl$="ROTATE Y"
RETURN

Twirl$="ROTATE X"
RETURN

Twirl$="ROTATE Z"
RETURN

Twirl$="TRANSLATE"
RETURN

Twirl$="TRANSLATE"
RETURN
210 Trans_ Z:!
7220 Twrl$="TRANSLATE"
7230 DISP " MODE: TRANSLATE Z"
7240 Way=3
7250 RETURN
7260 !
7270 Vector:!
7280 DISP " MODE: VECTOR TRANSLATION"
7290 CALL Vector(Skitter(*),Skitmod(*),Temp(*),Tempa(*),Total(*),Trans(*),Sys_trans_x,Sys_trans_y,Sys_trans_z,Newskit(*),Screen_x,Screen_y,Printflag$)
7300 RETURN
7310 !
7320 Angle:!
7330 DISP " MODE:INPUT ANGLE ROTATION"
7340 CALL Angle(Skitter(*),Skitmod(*),Temp(*),Tempa(*),Total(*),Trans(*),Sys_rot_x,Sys_rot_y,Sys_rot_z,Newskit(*),Screen_x,Screen_y,Printflag$)
7350 RETURN
7360 !
7370 !
7380 Plot:!
7390 CALL Plot_it(Plot_device)
7400 PLOTTER IS Plot_device,"HPGL"
7410 DISP "PLOT BEING GENERATED"
7420 CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y)
7430 PLOTTER IS CRT,"INTERNAL"
7440 PRINT "IN:RO;IP;UP;SP0;"
7450 PRINTER IS CRT
7460 BEEP 1464.84,.5
7470 DISP " PLOT FINISHED"
7480 GOTO Menu
7490 !
7500 Dump:!
7510 DISP "GRAPHICS DUMP BEING GENERATED"
7520 DUMP DEVICE IS Dump_device
7530 DUMP GRAPHICS
7540 PRINT CHR$(12)
7550 PRINTER IS Dump_device
7560 PRINT CHR$(12)
7570 BEEP 1464.84,.5
7580 DISP " GRAPHICS DUMP FINISHED"
7590 GOTO Menu
7600 !
7610 Printer:!
7620 DISP "WHERE IS THE LOCATION OF THE EXTERNAL PRINTER";
7630 LINPUT Temps
7640 IF Temp$="" THEN GOTO 7540
7650 Dump_device=VAL(Temp$)
7660 DISP " PRINTER IS AT ";CHR$(129);Dump_device;CHR$(128)
7670 GOTO Menu
7680 !
7690 Plotter:!
7700 DISP " WHERE IS THE LOCATION OF THE PLOTTER";
7710 LINPUT Temp$;
7720 IF Temp$="" THEN GOTO 7620
7730 Plot_device=VAL(Temp$)
7740 DISP " PLOTTER IS AT ";CHR$(129);Plot_device;CHR$(128)
7750 GOTO Menu
7760 !
7770 !**************************************************************************
7780 ! INITIALIZATION OF PARAMETERS
DEG
GINIT
GRAPHICS ON
PLOTTER IS CRT,"INTERNAL"

Dump_device=9
Plot_device=705

READ Skitter(*)

MAT Skitmod= Skitter
MAT Femurtemp= (1)
MAT Tibiatemp= (1)
MAT Trans= IDN
MAT Tempo= (1)
MAT Total= IDN
Menu$="MAIN"

Twirl$="ROTATE Y"
Freeleg$= "FIXED"
Way=1

Printflag$= "OFF"

Increment=5
Rot_increment=3

Screen_x=1
Screen_y=2

Sys_trans_x=0
Sys_trans_y=0
Sys_trans_z=0
Sys_rot_x=0
Sys_rot_y=0
Sys_rot_z=0

Fem_a_ang=0
Fem_b_ang=0
Fem_c_ang=0
Tib_a_ang=90
Tib_b_ang=90
Tib_c_ang=90
Pivot_ang_a=0
Pivot_ang_b=0
Pivot_ang_c=0

SET TO DEGREES
INITIATE GRAPHICS
TURN G-PLANE ON
INIT PLOTTER

READ SKITTER DATA

INIT TRANS MATRIX TO IDN

INIT MAIN MENU

AXIS OF TRANS X=1,Y=2,Z=3

TRANS INC.

INIT VIEW PLANE
X=1,Y=2,Z=3

INIT POSITIONS OF SYSTEM

INIT LEG ANGLES
Screen_x = Screen_x_min
Screen_y = Screen_y_max

SHOW Screen_x, Screen_x_max, Screen_y, Screen_y_max

RETURN

EXIT ROUTINE TO CLEAR SCREEN AND ENTER BASIC ENVIRONMENT

Finished:  
DONE WITH PROGRAM

GRAPHICS OFF
CLEAR SCREEN
END

SUBROUTINE ROTATE: ROTATES SYSTEM ABOUT LOCAL Y AXIS

SUB Rotate_y(Skitter(*), Skitmod(*), Total(*), Trans(*), Temp(*), Sys_rot_y, Theta, Printflag$)

DIM Bogus(4,4)

Sys_rot_y = Sys_rot_y + Theta

IF Printflag$="ON" THEN
  DISP "MODE: ROTATE Y BY ANGLE OF ", Theta, " TOTAL ANGLE=" , Sys_rot_y
END IF

MAT Tempa(*,1:3) = Skitter(*,1:3)

SET UP ROTATION MATRIX

MAT Trans = IDN
Sine = SIN(Theta)
Cosine = COS(Theta)
Trans(1,1) = Cosine
Trans(1,3) = Sine
Trans(3,1) = -Sine
Trans(3,3) = Cosine

INCREMENT Y ROTATION VARIABLE
SUBROUTINE ROTATE X: ROTATES SYSTEM ABOUT LOCAL X AXIS

SUB Rotate_x(Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Tempa(*),Sys_rot_x,Theta,Printflag$)

OPTION BASE 1
DIM Bogus(4,4)

Sys_rot_x=Sys_rot_x+Theta
IF Printflag$="ON" THEN
   DISP " MODE: ROTATE X BY ANGLE OF ",Theta," TOTAL ANGLE="
END IF

MAT Tempa(*,1:3)= Skitter(*,1:3)

SET UP ROTATION MATRIX
MAT Trans= IDN
Sine=SIN(Theta)
Cosine=COS(Theta)
Trans(2,2)=Cosine
Trans(3,3)=Cosine
Trans(2,3)=Sine
Trans(3,2)=Sine

INCREMENT ROTATION VARIABLE
MAT Bogus= Trans*Total
MAT Total= Bogus
MAT Temp= Tempa*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)

SUBEND

SUBROUTINE ROTATE Z: ROTATES SYSTEM ABOUT LOCAL Z AXIS

SUB Rotate_z(Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Tempa(*),Sys_rot_z,Theta,Printflag$)

OPTION BASE 1
DIM Bogus(4,4)

Sys_rot_z=Sys_rot_z+Theta
IF Printflag$="ON" THEN
   DISP " MODE: ROTATE Z BY ANGLE OF ",Theta," TOTAL ANGLE="
END IF

MAT Tempa(*,1:3)= Skitter(*,1:3)

SET UP ROTATION MATRIX

MAT Trans= IDN
Sine=SIN(Theta)
Cosine=COS(Theta)
Trans(1,1)=Cosine
Trans(1,2)=-Sine
Trans(2,1)=Sine
Trans(2,2)=Cosine

INCREMENT COUNTER; FIND NEW SKITTER MATRIX

MAT Bogus= Trans*Total
MAT Total= Bogus
MAT Temp= Tempa*Total
MAT Skltmod(*,1:3)= Temp(*,1:3)

SUBEND

**************************************************************************
* SUBROUTINE TRANSLATE 3D: TRANSLATES SYSTEM ALONG A LOCAL X-Y-Z AXIS *
**************************************************************************

SUB Translate_3d(Skitter(*),Skltmod(*),Total(*),Temp(*),Tempa(*),Trans(*),Sys_trans_x,Sys_trans_y,Sys_trans_z,Way,Theta,Printflag$)

OPTION BASE 1
DIM Laurie(4,4)

SET UP TEMP ARRAY SO AS NOT TO LOOSE PENS --SKITTER(*,4)

MAT Transa(*,1:3)= Skitter(*,1:3)

DETERMINE DIRECTION OF TRANSLATION AND SET UP TRANS MATRIX

IF Way=1 THEN
Sys_trans_x=Sys_trans_x+Theta
Tx=Theta
Ty=0
Tz=0
IF Printflag$="ON" THEN
DISP " MODE: TRANSLATE X BY ",Theta,"IN. TOTAL TRANSLATION=" ,Sys_trans_x
END IF
END IF
IF Way=2 THEN
Sys_trans_y=Sys_trans_y+Theta
Tx=0
Ty=Theta
Tz=0
IF Printflag$="ON" THEN
DISP " MODE: TRANSLATE Y BY ",Theta,"IN. TOTAL TRANSLATION=" ,Sys_trans_y
END IF
END IF
IF Way=3 THEN
Sys_trans_z=Sys_trans_z+Theta
Tx=0
Ty=0
Tz=Theta
END IF
3680 IF Printflag$="ON" THEN
3690 DISP " MODE: TRANSLATE Z BY ";Theta,"IN. TOTAL TRANSLATION=";Sys_t
3700 END IF
3710 END IF
3720 MAT Trans= IDN
3730 Trans(4,1)=Tx
3740 Trans(4,2)=Ty
3750 Trans(4,3)=Tz
3760 FIND NEW SKITTER MATRIX WITH CORRECT PENS
3770 MAT Laurie= Trans*Total
3780 MAT Total= Laurie
3790 MAT Temp= Tempa*Total
3800 MAT Skitmod(*,1:3)= Temp(*,1:3)
3810 SUBEND
3820*************************************************************************
3830 !*************************************************************************
3840 SUB Scaling_3d(Sx,Sy,Sz,Array(*))
3850 MAT Array= IDN
3860 Array(1,1)=Sx
3870 Array(2,2)=Sy
3880 Array(3,3)=Sz
3890 SUBEND
3900*************************************************************************
3910 !*************************************************************************
3920 SUBROUTINE DISPLAY_SKIT: PLOTS SKITTER TO LOCAL PLOTTING DEVICE OR SCREEN
3930*************************************************************************
3940 SUB Display_skit(Skitter(*),Newskit(*),Screen_x,Screen_y)
3950 OPTION BASE 1
3960 DATA 1,1,4  ! PEN1
3970 DATA 4,4,4  ! PEN 2
3980 DATA 8,8,4  ! PEN 3
3990 DIM Pen1(1,3),Pen2(1,3),Pen3(1,3),Temp(1,3)
4000 READ Pen1(*),Pen2(*),Pen3(*)
4010 IF Screen_x<>1 AND Screen_y<>2 THEN
4020 MAT Temp= Pen2
4030 MAT Pen2= Pen3
4040 MAT Pen3= Temp
4050 END IF
4060 MAT Newskit(*,1)= Skitter(*,Screen_x)
4070 MAT Newskit(*,2)= Skitter(*,Screen_y)
4080 MAT Newskit(*,3)= Skitter(*,4)
4090 MAT Newskit(37:37,*)= Pen1
4100 MAT Newskit(68:68,*)= Pen2
4110 MAT Newskit(99:99,*)= Pen3
4120 CLEAR SCREEN
4130 GCLEAR
4140 FRAME
4150 IF Screen_x<>3 AND Screen_y<>2 THEN GOTO 10300
4160 MOVE -100,-.9
4170 RECTANGLE 200,.15
4180 MOVE -100,-.75
4190 RECTANGLE 200,.15
MOVE -100,-.6
RECTANGLE 200,.15
MOVE -100,-.45
RECTANGLE 200,.15
MOVE -100,-.3
RECTANGLE 200,.15
MOVE -100,-.15
RECTANGLE 200,.15
PLOT Nevskit(*)
LINE TYPE 1
SUBEND

SUB Printmat(Array(*))
OPTION BASE 1
FOR Row=BASE(Array,1) TO SIZE(Array,1)+BASE(Array,1)-1
    FOR Column=BASE(Array,2) TO SIZE(Array,2)+BASE(Array,2)-1
        PRINT USING "DDDD.DD,XX,#";Array(Row,Column)
    NEXT Column
PRINT
NEXT Row
SUBEND

SUBROUTINE TRANS_TO_VECTOR: TRANSLATES SYSTEM TO A GIVEN POINT

SUB Trans_to_vector(Skitter(*),Skitmod(*),Temp(*),Tempa(*),Total(*),Trans(*))
OPTION BASE 1
DIM Bogus(4,4)
MAT Tempa(*,1:3)= Skitter(*,1:3)
MAT Trans= IDN
MAT Bogus= Trans*Total
MAT Total= Bogus
MAT Temp= Tempa*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)
SUBEND

SUBROUTINE VECTOR: TRANSLATES SYSTEM TO INPUT POINT
10744 !
10750 SUB Vector(Skitter(*),Skitmod(*),Temp(*),Tempa(*),Total(*),Trans(*),Sys_trans_x,Sys_trans_y,Sys_trans_z,Newskit(*),Screen_x,Screen_y,Prinflag$)
10760 !
10770 ! ASK FOR INPUT <CR> MEANS LEAVE
10780 !
10790 Ask:
10800 DISP " X COORDINANT RELATIVE TO ",Sys_trans_x;
10810 INPUT Temp$
10820 IF Temp$="" THEN
10830 GOTO Leave
10840 ELSE
10850 X-VAL(Temp$)
10860 END IF
10870 DISP " Y COORDINANT RELATIVE TO ",Sys_trans_y;
10880 INPUT Temp$
10890 IF Temp$="" THEN
10900 GOTO Leave
10910 ELSE
10920 Y-VAL(Temp$)
10930 END IF
10940 DISP " Z COORDINANT RELATIVE TO ",Sys_trans_z;
10950 INPUT Temp$
10960 IF Temp$="" THEN
10970 GOTO Leave
10980 ELSE
10990 Z-VAL(Temp$)
11000 END IF
11010 ! UPDATE TRANSLATION COUNTERS
11020 !
11030 !
11040 Sys_trans_x-Sys_trans_x+X
11050 Sys_trans_y-Sys_trans_y+Y
11060 Sys_trans_z-Sys_trans_z+Z
11070 !
11080 ! FIND NEW SKITTER MATRIX
11090 !
11100 CALL Trans_to_vector(Skitter(*),Skitmod(*),Temp(*),Tempa(*),Total(*),Trans(*),X,Y,Z)
11110 !
11120 ! DISPLAY SKITTER
11130 CALL Display_sklt(Skitmod(*),Newskit(*),Screen_x,Screen_y)
11140 GOTO Ask
11150 Leave:!
11160 SUBEND
11170 !
11180 !
11190 !******************************************************************************
11200 !
11210 ! SUBROUTINE WINDOWS: Allows user to change viewing axis
11220 !
11230 !******************************************************************************
11240 !
11250 SUB Windows(Skitmod(*),Newskit(*),Screen_x,Screen_y)
11260 !
11270 IF Screen x=1 AND Screen_y=2 THEN
11280 !
11290 END IF
11300 IF Screen x=1 AND Screen_y=3 THEN
11310 !
11320 END IF
```
1280 IF Screen_x=3 AND Screen_y=2 THEN
1290   DISP " CURRENT WINDOW = Z-Y PLANE"
1300 END IF
1310 Menu:
1320 ON KEY 0 LABEL "X AXIS" GOTO Zy_plane
1330 ON KEY 2 LABEL "Y AXIS" GOTO Xz_plane
1340 ON KEY 4 LABEL "Z AXIS" GOTO Xy_plane
1350 ON KEY 9 LABEL "QUIT" GOTO Leave
1360 GOTO 11360
1370
1380 Zy_plane:
1390   Screen_x=3
1400   Screen_y=2
1410   DISP " NEW WINDOW = LOOKING DOWN X AXIS"
1420   CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y
1430   GOTO Menu
1440
1450 Xz_plane:
1460   Screen_x=1
1470   Screen_y=3
1480   DISP " NEW WINDOW = LOOKING DOWN Y AXIS"
1490   CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y
1500   GOTO Menu
1510
1520 Xy_plane:
1530   Screen_x=1
1540   Screen_y=2
1550   DISP " NEW WINDOW = LOOKING DOWN Z AXIS"
1560   CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y
1570   GOTO Menu
1580 Leave:
1590 SUBEND
1600 !
1610!
1620!
1630!***********************************************************************************************
1640! SUBROUTINE WINDOW_LIMITS: ALLOWS THE USER TO INPUT NEW VIEWING WINDOW
1650 SUB Window_limits(Skitmod(*),Newskit(*),Screen_x,Screen_y,Screenx_win_min, Screenx_win_max, Screeny_win_min, Screeny_win_max)
1660!
1670!
1680!
1690 DISP " INPUT XMIN --- CURRENT VALUE IS",Screenx_win_min," <CR> TO EXIT";
1700 LINPUT Temp$:
1710 IF Temp$="" THEN GOTO Leave
1720 DISP " INPUT XMAX --- CURRENT VALUE IS",Screenx_win_max," <CR> TO EXIT";
1730 LINPUT Temp1$
1740 IF Temp1$="" THEN GOTO Leave
1750 IF VAL(Temp$)>VAL(Temp1$) THEN
1760   BEEP 1464.84,.5
1770   DISP " XMIN HAS TO BE LESS THAN XMAX"
1780   WAIT 3
1790   GOTO 11690
```
ELSE
Screenx_win_min=VAL(Temp$)
Screenx_win_max=VAL(Temp1$)
END IF

1910 IF Temp$="" THEN GOTO Leave
1920 END IF
1930 IF VAL(Temp$)>VAL(Temp1$) THEN
1940 BEEP 146484,.5
1950 DISP " Y MIN MUST BE LESS THAN Y MAX"
1960 WAIT 3
1970 GOTO 11870
1980 ELSE
Screeny_win_min=VAL(Temp$)
Screeny_win_max=VAL(Temp1$)
1990 END IF
2000 SHOW Screenx_win ,Screenx_win_max,Screenx_win_min,Screenx_win_max
2010 CALL Display_skit (Skitter(*),Newskit(*),Screen_x,Screen_y)
2020 END
2030 !
2040 SHOW Screeny_win,min,Screeny_win_max,Screeny_win_min,Screeny_win_max
2050 CALL Display_skit (Skitter(*),Newskit(*),Screen_x,Screen_y)
2060 !
2070 Leave:!
2080 SUBEND
2090 !
2100 !*************************************************************************
2110 !
2120 ! SUBROUTINE ANGLE: ALLOWS USER TO ROTATE SYSTEM BY INPUT ANGLE
2130 ! ABOUT LAST ROTATION AXIS
2140 !*************************************************************************
2150 !
2160 SUB Angle(Skitter(*),Skitmod(*),Temp(*),Tempa(*),Total(*),Trans(*),Sys_rot_x,
Sys_rot_y,Sys_rot_z,Newskit(*),Screen_x,Screen_y,Printflag$,$Twirl$)
2170 !
2180 ! FIND MODE AND ROTATE ABOUT CORRECT AXIS
2190 !
2200 IF Twirl$="ROTATE X" THEN
2210 DISP " INPUT ANGLE TO ROTATE ABOUT X AXIS -- CURRENT ANG="",Sys_rot_x;
2220 LINPUT Temp$
2230 IF Temp$="" THEN GOTO Leave
2240 CALL Rotate_x (Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Tempa(*),Sys_rot_x,Theta,Printflag$)
2250 END IF
2260 !
2270 !
2280 IF Twirl$="ROTATE Y" THEN
2290 DISP " INPUT ANGLE TO ROTATE ABOUT Y AXIS -- CURRENT ANG="",Sys_rot_y;
2300 LINPUT Temp$
2310 IF Temp$="" THEN GOTO Leave
2320 Theta=VAL(Temp$)
2330 CALL Rotate_y (Skitter(*),Skitmod(*),Total(*),Trans(*),Temp(*),Tempa(*),Sys_rot_y,Theta,Printflag$)
2340 END IF
2350 !
2360 !
2370 !*************************************************************************
12330!  
12340 IF Twirl$="ROTATE Z" THEN  
12350 DISP " INPUT ANGLE TO ROTATE ABOUT Z AXIS -- CURRENT ANG=", Sys_rot_z;  
12360 LINPUT Temp$  
12370 IF Temp$="" THEN GOTO Leave  
12380 Theta=VAL(Temp$)  
12390 CALL Rotate_z(Skitter(*), Skitmod(*), Total(*), Trans(*), Temp(*), Tempa(*), Sys_rot_z, Theta, Printflag$)  
12400 END IF  
12410 ! OUTPUT NEW PICTURE  
12420 !  
12430 !  
12440 CALL Display_skit(Skitmod(*), Newskit(*), Screen_x, Screen_y)  
12450 GOTO 12150  
12460 !  
12470 !  
12480 Leave:!  
12490 ! SUBEND  
12500 !  
12510 ! SUBROUTINE WHAT: OUTPUTS PROGRAM VARIABLES TO SCREEN  
12513 !  
12514 !***************************************************************************  
12520 !  
12530 SUB What(Sys_trans_x, Sys_trans_y, Sys_trans_z, Sys_rot_x, Sys_rot_y, Sys_rot_z, A, B, C, D, Printflag$, Screenx, Screeny, Plotd, Dumpd, Fa, Fb, Fc, Ta, Tb, Tc, Inc, Sp, Tp, Up)  
12540 !  
12550 GRAPHICS OFF  
12560 CLEAR SCREEN  
12570 PRINT CHR$(132);"THESE ARE THE CURRENT PARAMETERS:";CHR$(128)  
12580 PRINT  
12590 PRINT CHR$(129);" WINDOW: ";CHR$(128);" X MIN = ";A;" X MAX = ";B  
12600 PRINT " Y MIN = ";C;" Y MAX = ";D  
12610 PRINT  
12620 IF Screenx=1 AND Screeny=2 THEN PRINT CHR$(129);" VIEW: ";CHR$(128);" LOOKING DOWN Z AXIS"  
12630 IF Screenx=3 AND Screeny=2 THEN PRINT CHR$(129);" VIEW: ";CHR$(128);" LOOKING DOWN X AXIS"  
12640 IF Screenx=1 AND Screeny=3 THEN PRINT CHR$(129);" VIEW: ";CHR$(128);" LOOKING DOWN Y AXIS"  
12650 PRINT  
12660 PRINT "PLOTTER LOCATION:";CHR$(129);Plotd;CHR$(128);" PRINTER LOCATION" N=";CHR$(129);Dumpd;CHR$(128)  
12670 PRINT  
12680 IF Printflag$="ON" THEN  
12690 PRINT " DISPLAY VALUES IS ";CHR$(129);" ON ";CHR$(128)  
12700 ELSE  
12710 PRINT " DISPLAY VALUES IS OFF"  
12720 END IF  
12730 PRINT  
12740 PRINT TAB(5);CHR$(132);"TRANS (in.) ROT (deg)";CHR$(128);" FEMUR ANGLE TIBIA ANGLE";CHR$(128)  
12750 PRINT  
12760 Brice: IMAGE AA, 2X, 4D.2D, 6X, 4D.2D, 18X, A, 6X, 4D.2D, 8X, 4D.2D  
12770 PRINT USING Brice:"X ";Sys_trans_x;Sys_rot_x;"A";Fa;Ta  
12780 PRINT USING Brice:"Y ";Sys_trans_y;Sys_rot_y;"B";Fb;Tb  
12790 PRINT USING Brice:"Z ";Sys_trans_z;Sys_rot_z;"C";Fc;Tc
PRINT "PIVOT ANG A:";Sp;"  PIVOT ANG B:";Tp;"  PIVOT ANG C:";Up
INPUT "HIT <CR> TO CONTINUE",Temp$
CLEAR SCREEN
ALPHA OFF
GRAPHICS ON
SUBEND

SUBROUTINE ROTATE LEG Z: ROTATES ENTIRE LEG ABOUT Z AXIS

SUB Rotate_leg_z(Femur(*),Femurmod(*),Leg$,Theta,Printflag$,OPTIONAL Femur a_rot,Way)
OPTION BASE 1
DIM Bogus(4,4),Temp(196,4),Tempa(196,4),Trans(4,4)
N=SIZE(Femur,1)
REDim Tempa(N,4),Temp(N,4)

IF Way=1 THEN
Femura_rot=Femura_rot+Theta

IF Printflag$="ON" THEN
DISP " MODE: ";Leg$;" THROUGH ANGLE OF ";Theta;" TOTAL ANGLE=";Femura_rot
END IF
END IF
MAT Tempa=(1)
MAT Tempa(*,1:3)= Femur(*,1:3)

SET UP ROTATION MATRIX
MAT Trans= IDN
Sine=SIN(Theta)
Cosine=COS(Theta)
Trans(1,1)=Cosine
Trans(1,2)=Sine
Trans(?1)=Sine
Trans(2,2)=Cosine

FIND NEW SKITTER MATRIX
MAT Temp= Tempa*Trans
MAT Femurmod(*,1:3)= Temp(*,1:3)
SUBEND

SUBROUTINE TRANS FEM ORIG: TRANSLATES LEG TO ORIGIN

SUB Trans_fem_orig(Femur(*),Femurmod(*),X,Y,Z)
OPTION BASE 1
DIM Bogus(4,4),Temp(196,4),Temp(196,4),Trans(4,4)
N=SIZE(Femur,1)
13300 REDIM Tempa(N,4),Temp(N,4)
13310 | SET UP STORAGE ARRAY TO KEEP SKITTER PENS CORRECT SKITTER(*,4)
13330 |
13340 | MAT Tempa= (1)
13350 | MAT Tempa(*,1:3)= Femur(*,1:3)
13360 | MAT Trans= IDN
13370 |
13380 | SET UP TRANS MATRIX
13390 |
13400 | Trans(4,1)=X
13410 | Trans(4,2)=Y
13420 | Trans(4,3)=Z
13430 |
13440 | FIND NEW MATRIX
13450 |
13460 | MAT Temp= Tempa*Trans
13470 | MAT Femurmod(*,1:3)= Temp(*,1:3)
13480 SUBEND
13490 |
13500 |*******************************************************************************
13510 |
13511 | SUBROUTINE ROTATE LEG Y: ROTATES LEG ABOUT Y AXIS
13512 |
13513 |*******************************************************************************
13514 |
13520 | SUB Rotate_leg_y(Femur(*),Femurmod(*),Theta)
13530 |
13540 | OPTION BASE 1
13550 | DIM Bogus(4,4),Temp(196,4),Tempa(196,4),Trans(4,4)
13560 | N=SIZE(Femur,1)
13570 | REDIM Tempa(N,4),Temp(N,4)
13580 |
13590 | MAT Tempa= (1)
13600 | MAT Tempa(*,1:3)= Femur(*,1:3)
13610 |
13620 | SET UP ROTATION MATRIX
13630 |
13640 | MAT Trans= IDN
13650 | Sine=SIN(Theta)
13660 | Cosine=COS(Theta)
13670 | Trans(1,1)=Cosine
13680 | Trans(1,3)=Sine
13690 | Trans(3,1)=Sine
13700 | Trans(3,3)=Cosine
13710 |
13720 | FIND NEW SKITTER MATRIX
13730 |
13740 | MAT Temp= Tempa*Trans
13750 | MAT Femurmod(*,1:3)= Temp(*,1:3)
13760 | SUBEND
13770 |
13780 |
13790 |*******************************************************************************
13800 |
13801 | SUBROUTINE ZOOM PAN: ALLOWS USER TO PAN OR ZOOM WINDOW
13802 |
13803 |*******************************************************************************
13804 |
13810 SUB Zoom_pan(Window$,Xmin,Xmax,Ymin,Ymax,Skitmod(*),Newkit(*),Screen_x,Sc
ON ERROR GOTO Brice

L3820 DISP " YOUR CURRENT WINDOW VALUES ARE XMIN:";Xmin;" XMAX:";Xmax;" YM N:";Ymin;" YMAX:";Ymax
L3830 Menu: 1
L3840 ON KNOB .15 GOTO Knob_isr
L3850 ON KEY 9 LABEL "QUIT" GOTO Leave
L3860 ON KEY 0 LABEL "ZOOM" GOTO Zoom
L3870 ON KEY 1 LABEL "" GOTO 13970
L3880 ON KEY 2 LABEL "PAN X" GOTO Pan_x
L3890 ON KEY 3 LABEL "" GOTO 13970
L3900 ON KEY 4 LABEL "PAN Y" GOTO Pan_y
L3910 ON KEY 5 LABEL "" GOTO 13970
L3920 ON KEY 6 LABEL "INPUT DATA " GOTO Input_data
L3930 ON KEY 7 LABEL "" GOTO 13970
L3940 ON KEY 8 LABEL "" GOTO 13970
L3950 GOTO 13970
L3960 Pan_x:
L3970 Window$="PAN X"
L3980 GOTO Menu
L3990 Pan_y:
L4000 Window$="PAN Y"
L4010 GOTO Menu
L4020 Zoom:
L4030 Window$="ZOOM"
L4040 GOTO Menu
L4050 Knob_isr:
L4060 Th et a=KNOBx
L4070 IF Window$="ZOOM" AND Theta>0 THEN
L4080 Xmin=Xmin-5
L4090 Xmax=Xmax+5
L4100 Ymin=Ymin-5
L4110 Ymax=Ymax+5
L4120 SHOW Xmin,Xmax,Ymin,Ymax
L4130 END IF
L4140 IF Window$="ZOOM" AND Theta<0 THEN
L4150 Xmin=Xmin+5
L4160 Xmax=Xmax-5
L4170 Ymin=Ymin+5
L4180 Ymax=Ymax-5
L4190 SHOW Xmin,Xmax,Ymin,Ymax
L4200 END IF
L4210 IF Window$="PAN X" AND Theta<0 THEN
L4220 Xmin=Xmin+5
L4230 Xmax=Xmax-5
L4240 Ymin=Ymin+5
L4250 Ymax=Ymax-5
L4260 SHOW Xmin,Xmax,Ymin,Ymax
L4270 END IF
L4280 IF Window$="PAN X" AND Theta>0 THEN
L4290 Xmin=Xmin-5
L4300 Xmax=Xmax+5
L4310 SHOW Xmin,Xmax,Ymin,Ymax
L4320 END IF
L4330 IF Window$="PAN X" AND Theta>0 THEN
L4340 Xmin=Xmin-5
L4350 Xmax=Xmax+5
L4360 SHOW Xmin,Xmax,Ymin,Ymax
L4370 END IF
END IF

IF Window$="PAN Y" AND Theta>0 THEN
  Ymin=Ymin+5
  Ymax=Ymax+5
  SHOW Xmin,Xmax,Ymin,Ymax
END IF

IF Window$="PAN Y" AND Theta<0 THEN
  Ymin=Ymin-5
  Ymax=Ymax-5
  SHOW Xmin,Xmax,Ymin,Ymax
END IF

CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y)
GOTO 13840

Input data:

DISP " INPUT XMIN --- CURRENT VALUE IS",Xmin," <CR> TO EXIT";
INPUT Temp$
IF Temp$="" THEN GOTO Leave
DISP " INPUT XMAX --- CURRENT VALUE IS",Xmax," <CR> TO EXIT";
INPUT Temp$
IF Temp$="" THEN GOTO Leave
IF VAL(Temp$) > VAL(Temp$) THEN
  BEEP 1464.84,.5
  DISP " XMIN HAS TO BE LESS THAN XMAX"
  WAIT 3
  GOTO 14610
ELSE
  Xmin=VAL(Temp$)
  Xmax=VAL(Temp$)
END IF

DISP " INPUT YMIN --- CURRENT VALUE IS",Ymin," <CR> TO EXIT";
INPUT Temp$
IF Temp$="" THEN GOTO Leave
DISP " INPUT YMAX --- CURRENT VALUE IS",Ymax," <CR> TO EXIT";
INPUT Temp$
IF Temp$="" THEN GOTO Leave
IF VAL(Temp$) > VAL(Temp$) THEN
  BEEP 1464.84,.5
  DISP " Y MIN MUST BE LESS THAN Y MAX"
  WAIT 3
  GOTO 14790
ELSE
  Ymin=VAL(Temp$)
  Ymax=VAL(Temp$)
END IF

SHOW Xmin,Xmax,Ymin,Ymax
CALL Display_skit(Skitmod(*),Newskit(*),Screen_x,Screen_y)
GOTO Menu
I_15000  
Brice:  
I I F 5010 ERRN-31 THEN  
15040  
15050  
15060  
15070  
15080  
15090  
15100  
15110  
15120  
15130  
15140  SUBEND  
15160 !  
15161 ! SUBROUTINE PLOT IT: SETS UP PLOTTER PARAMETERS FOR HARD COPY  
15162 !  
15163 !******************************************************************************  
15164 !  
15170 SUB Plot_it(Plot_device)  
15180 DIM L$[32]  
15190 DISP " HOW MANY QUADRANTS 1,2 OR 4 DEFAULT=1";  
15200 LINPUT Quad$  
15210 IF Quad$="" THEN  
15220 Quad=1  
15230 ELSE  
15240 Quad=VAL(Quad$)  
15250 END IF  
15260 IF Quad=1 THEN GOTO Label  
15270 IF Quad<>1 AND Quad<>2 AND Quad<>4 THEN  
15280 BEEP 1464,.5  
15290 GOTO 15190  
15300 END IF  
15310 DISP " WHICH SQUARE DEFAULT=1";  
15320 LINPUT What$  
15330 IF What$="" THEN  
15340 What=1  
15350 ELSE  
15360 What=VAL(What$)  
15370 END IF  
15380 IF What<>1 AND What<>2 AND What<>3 AND What<>4 THEN  
15390 BEEP 1464,.5  
15400 GOTO 15310  
15410 END IF  
15420 !  
15430 !  
15440 Label: !  
15450 DISP " INPUT ANY LABELS DEFAULT= NONE";  
15460 LINPUT L$  
15470 IF L$="" THEN GOTO P1  
15480 P1: !  
15490 PRINTER IS Plot_device  
15500 IF Quad=1 THEN  
15510 PRINT "IN;IP;SP1;SI .5,.5;PA 425,1000,lb”;L$;CHR$(3);";SP0;"  
15520 END IF  
15530 IF Quad=4 THEN  
15540 IF What=1 THEN  
15550 PRINT "IN;IP250,596,5250,4196;SP1;SI;PA 425,900,lb”;L$;CHR$(3);";S  
15560 END
IF What-2 THEN
PRINT "IN;IP5250,596,10250,4196;SP1;SI;PA 5425,900,LP";L$;CHR$(3);";SP0;"
END IF
IF What-3 THEN
PRINT "IN;IP250,4196,5250,7796;SP1;SI;PA 425,4500,LP";L$;CHR$(3);";SP0;"
END IF
IF What-4 THEN
PRINT "IN;IP5250,4196,10250,7796;SP1,SI,PA 5425,4500,LP";L$;CHR$(3);";SP0;"
END IF
IF Quad-2 THEN
IF What=2 THEN
PRINT "IN;RO90;IP;IW;IP154,244,7354,5122;SP1;SI .35,.35;PA 600,600;LB";L$;CHR$(3);";SP0;"
ELSE
PRINT "IN;RO90;IP;IW;IP154,5122,7354,10244;SP1;SI .35,.35;PA 600,600,5478;LB";L$;CHR$(3);";SP0;"
END IF
END IF
END IF
END IF
END IF
END IF
SUBROUTINE MOVIE: ALLOWS USER TO INPUT DATA FILES FOR ANIMATION OF POSITION SEQUENCES
0,0,0,10  |  A
6.5482,28.4912,0,-2  |  A
9.1602,20.9817,3.2283,-1  |  H
-1.7468,20.9817,9.5688,-1  |  I
-3.2741,28.4912,5.6709,-1  |  B
0,0,0,7  
0,0,0,10
-3.2741,28.4912,5.6709,-2  |  B
-7.3759,20.9817,6.3188,-1  |  J
-7.3759,20.9817,-6.3188,-1  |  K
-3.2741,28.4912,-5.6709,-1  |  C
0,0,0,7  
0,0,0,10
-3.2741,28.4912,-5.6709,-2  |  C
-1.7468,20.9817,-9.5688,-1  |  L
9.1602,20.9817,-3.2283,-1  |  G
6.5482,28.4912,0,-1  |  A
0,0,0,7  
0,0,0,7
UPPER BODY
SUB Movie
<table>
<thead>
<tr>
<th>Address</th>
<th>Data 1</th>
<th>Data 2</th>
<th>Data 3</th>
<th>Data 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>15990</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16000</td>
<td>6.5482</td>
<td>13.4722</td>
<td>0.00.0</td>
<td>-2</td>
</tr>
<tr>
<td>16010</td>
<td>9.1602</td>
<td>20.9817</td>
<td>3.2283</td>
<td>-1</td>
</tr>
<tr>
<td>16020</td>
<td>-1.7468</td>
<td>20.9817</td>
<td>9.5688</td>
<td>-1</td>
</tr>
<tr>
<td>16030</td>
<td>-3.2741</td>
<td>13.4722</td>
<td>5.6709</td>
<td>-1</td>
</tr>
<tr>
<td>16040</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16050</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16060</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16070</td>
<td>-3.2741</td>
<td>13.4722</td>
<td>5.6709</td>
<td>-2</td>
</tr>
<tr>
<td>16080</td>
<td>-7.3759</td>
<td>20.9817</td>
<td>6.3188</td>
<td>-1</td>
</tr>
<tr>
<td>16090</td>
<td>-7.3759</td>
<td>20.9817</td>
<td>-6.3188</td>
<td>-1</td>
</tr>
<tr>
<td>16100</td>
<td>-3.2741</td>
<td>13.4722</td>
<td>-5.6709</td>
<td>-1</td>
</tr>
<tr>
<td>16110</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16120</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16130</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16140</td>
<td>-3.2741</td>
<td>13.4722</td>
<td>-5.6709</td>
<td>-2</td>
</tr>
<tr>
<td>16150</td>
<td>-7.3759</td>
<td>20.9817</td>
<td>-9.5688</td>
<td>-1</td>
</tr>
<tr>
<td>16160</td>
<td>9.1602</td>
<td>20.9817</td>
<td>-3.2283</td>
<td>-1</td>
</tr>
<tr>
<td>16170</td>
<td>6.5482</td>
<td>13.4722</td>
<td>0.00.0</td>
<td>-2</td>
</tr>
<tr>
<td>16180</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16190</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16200</td>
<td>0.00.0</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16210</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16230</td>
<td>29.6602</td>
<td>20.9817</td>
<td>-3.2283</td>
<td>-1</td>
</tr>
<tr>
<td>16240</td>
<td>18.7362</td>
<td>25.0731</td>
<td>0.00.0</td>
<td>-1</td>
</tr>
<tr>
<td>16250</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16260</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16270</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16280</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16290</td>
<td>29.6602</td>
<td>20.9817</td>
<td>3.2283</td>
<td>-2</td>
</tr>
<tr>
<td>16300</td>
<td>18.7362</td>
<td>25.0731</td>
<td>0.00.0</td>
<td>-1</td>
</tr>
<tr>
<td>16310</td>
<td>9.1602</td>
<td>20.9817</td>
<td>3.2283</td>
<td>-1</td>
</tr>
<tr>
<td>16320</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16330</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16340</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16360</td>
<td>29.6602</td>
<td>20.9817</td>
<td>-3.2283</td>
<td>-1</td>
</tr>
<tr>
<td>16370</td>
<td>20.571</td>
<td>16.8903</td>
<td>0.00.0</td>
<td>-1</td>
</tr>
<tr>
<td>16380</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16390</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16400</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16410</td>
<td>29.6602</td>
<td>20.9817</td>
<td>3.2283</td>
<td>-2</td>
</tr>
<tr>
<td>16420</td>
<td>20.571</td>
<td>16.8903</td>
<td>0.00.0</td>
<td>-1</td>
</tr>
<tr>
<td>16430</td>
<td>9.1602</td>
<td>20.9817</td>
<td>3.2283</td>
<td>-1</td>
</tr>
<tr>
<td>16440</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16450</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16460</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16470</td>
<td>29.6602</td>
<td>20.9817</td>
<td>-3.2283</td>
<td>-2</td>
</tr>
<tr>
<td>16480</td>
<td>30.4255</td>
<td>0.00.0</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>16490</td>
<td>35.7901</td>
<td>9.2132</td>
<td>0.00.0</td>
<td>-1</td>
</tr>
<tr>
<td>16500</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16510</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16520</td>
<td>0.00.0</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16530</td>
<td>29.6602</td>
<td>20.9817</td>
<td>3.2283</td>
<td>-2</td>
</tr>
<tr>
<td>16540</td>
<td>30.4255</td>
<td>0.00.0</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>16550</td>
<td>35.7901</td>
<td>9.2132</td>
<td>0.00.0</td>
<td>-1</td>
</tr>
<tr>
<td>16560</td>
<td>0.00.0</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16570</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16580</td>
<td>0.00.0</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA</td>
<td>16590</td>
<td>0,0,0,10</td>
<td>16600</td>
<td>-1.7468, 20.9817, 9.5688, -2</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>----------</td>
<td>-------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td></td>
<td>16650</td>
<td>0,0,0,10</td>
<td>16660</td>
<td>-17.6259, 20.9817, 24.0723, -2</td>
</tr>
<tr>
<td></td>
<td>16710</td>
<td>0,0,0,10</td>
<td>16720</td>
<td>-17.6259, 20.9817, 9.5688, -2</td>
</tr>
<tr>
<td></td>
<td>16770</td>
<td>0,0,0,10</td>
<td>16780</td>
<td>-17.6259, 20.9817, 24.0723, -2</td>
</tr>
<tr>
<td></td>
<td>16840</td>
<td>0,0,0,10</td>
<td>16850</td>
<td>-11.9968, 20.9817, 27.3223, -2</td>
</tr>
<tr>
<td></td>
<td>16890</td>
<td>0,0,0,10</td>
<td>16910</td>
<td>-17.6259, 20.9817, 24.0723, -2</td>
</tr>
<tr>
<td></td>
<td>16950</td>
<td>0,0,0,10</td>
<td>16960</td>
<td>0,0,0,4</td>
</tr>
<tr>
<td></td>
<td>16980</td>
<td>0,0,0,10</td>
<td>16990</td>
<td>-7.3759, 20.9817, -6.3188, -2</td>
</tr>
<tr>
<td></td>
<td>17010</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17020</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17030</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17040</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17050</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17060</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17070</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17080</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17090</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17100</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17110</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17120</td>
<td>0,0,0,10</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17130</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17140</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17150</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17160</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17170</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17180</td>
<td>0,0,0,7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**FEMUR TWO**

**TIBIA TWO**

**FEMUR THREE**

**RESERVED FOR PEN**
17190 DATA -1.7468,20.9817,-9.5588,-1 | L
17200 DATA 0,0,0,7
17210 |
17220 DATA 0,0,0,10 |
17230 DATA -17.6259,20.9817,-24.0723,-2 | U
17240 DATA -15.1940,0,-26.3601,-1 | E'
17250 DATA -17.8763,9.2132,-31.0059,-1 | F'
17260 DATA 0,0,0,7
17270 |
17280 DATA 0,0,0,10 |
17290 DATA -11.9968,20.9817,-27.3223,-2 | V
17300 DATA -15.1940,0,-26.3601,-1 | E'
17310 DATA -17.8763,9.2132,-31.0059,-1 | F'
17320 DATA 0,0,0,7
17330 REM ****************************************************************
17340 |
17350 |
17360 | OPTION BASE 1
17370 REAL Skitter(129,4),Newsklt(129,3) | DEFINE VAR
17380 REAL Trans(4,4),Temp(129,4),Tempa(129,4) | TRANSFORMATION MATRIX
17390 REAL Total(4,4),Skitmod(129,4) | TOTAL TRAFOFORM MATRIX
17400 REAL Femur(31,4),Femurmod(31,4),Femurtmp(31,4)
17410 REAL Tibia(10,4),Tibiastemp(10,4),Tibiamod(10,4)
17420 INTEGER Penl(l,3),Pen2(l,3),Pen3(l,3)
17430 GOSUB Init | INITIALIZATION ROUTINE
17440 CALL Display_skit(Skitter(*),Newsklt(*),Screen_x,Screen_y) | DRAW SKITTER
17450 |
17460 |
17470 DATA 1,1,4 | PEN1
17480 DATA 4,4,4 | PEN2
17490 DATA 8,8,4 | PEN3
17500 |
17510 Start:
17520 DISP "DO YOU WANT TO RUN AN ALREADY COMPUTED FILE  Y OR N";
17530 INPUT Ans$
17540 |
17550 |
17560 |
17570 | IF Ans$="Y" THEN
17580 | DISP "NAME OF COMPUTED FILE";
17590 | INPUT Name$
17600 | IF Name$="" THEN
17610 | GOTO 17520
17620 | ELSE
17630 | Skitwork$=Name$
17640 | File_flag=1
17650 | GOTO Movie
17660 | END IF
17670 | END IF
17680 |
17690 | IF Ans$="N" THEN
17700 | DISP "INPUT MOVIE FILE";
17710 | INPUT Files
17720 | IF Files="" THEN
17730 | GOTO 17700
17740 | ELSE
17750 | Skitwork$="SKITWORK"
17760 | END IF
17770 | CREATE BDAT Skitwork$,400
ASSIGN @Path1 TO Skitwork$
ASSIGN @Path2 TO File$
END IF
READ Pen1(*),Pen2(*),Pen3(*)
Top:
ENTER @Path2;Tim,Free,Fadat,Tadat,Fbdat,Tbdat,Fcdat,Tcdat,Xr,Yr,Zr,Tx
*.Ty,Tz

**************************************************************************

** DETERMINE TRANSFORMATION MATRICES FOR EACH TIME UNIT **

IF Tim=999 THEN GOTO Movie
N=N+1
IF Free=0 THEN Freeleg$="FIXED"
IF Free=1 OR Free=2 THEN Freeleg$="FREE"

IF Fadat<>0 AND Fadat<>999 THEN
Leg$="FEMUR A"
IF Free=2 THEN
Leg_flag$="FEMUR A"
END IF
Theta=Fadat
GOSUB Leg_isr

IF Fadat=999 THEN
Leg_flag$="FEMUR A"
END IF

IF Tadat<>0 THEN
Leg$="TIBIA A"
Theta=Tadat
GOSUB Leg_isr

IF Fbdat<>0 AND Fbdat<>999 THEN
Leg$="FEMUR B"
IF Free=2 THEN
Leg_flag$="FEMUR B"
END IF
Theta=Fbdat
GOSUB Leg_isr

IF Fbdat=999 THEN
Leg_flag$="FEMUR B"
END IF

IF Tbdat<>0 THEN
Leg$="TIBIA B"
Theta=Tbdat
GOSUB Leg_isr

IF Fcdat<>0 AND Fcdat<>999 THEN
Leg$="FEMUR C"
IF Free=2 THEN
  Leg_flag$="FEMUR C"
END IF

Theta=Fcdat
GOSUB Leg_isr

IF Fcdat=999 THEN
  Leg_flag$="FEMUR C'
END IF

IF Tcdat<>O THEN
  Leg$="TIBIA C"
  Theta=Tcdat
  GOSUB Leg_isr
END IF

IF Yr<>0 THEN
  Twirl$="ROTATE Y"
  Theta=Yr
  GOSUB System_isr
END IF

IF Xr<>0 THEN
  Twirl$="ROTATE X"
  Theta=Xr
  GOSUB System_isr
END IF

IF Zr<>0 AND Free<>2 THEN
  Twirl$="ROTATE Z"
  Theta=Zr
  GOSUB System_isr
END IF

IF Zr<>0 AND Free=2 THEN
  Twirl$="PIVOT"
  Theta=Zr
  GOSUB System_isr
END IF

IF Tx<>0 THEN
  Way=1
  Theta=Tx
  Twirl$="TRANSLATE"
  CALL Printmat(Total(*))
  GOSUB System_isr
END IF

IF Ty<>0 THEN
  Way=2
  Theta=Ty
  Twirl$="TRANSLATE"
  GOSUB System_isr
END IF

IF Tz<>0 THEN
  Way=3
  Theta=Tz
  Twirl$="TRANSLATE"
  GOSUB System_isr
MAT Tempa(*,1:3) = Skitter(*,1:3)
MAT Temp = Temp*Total
MAT Skitmod(*,1:3) = Temp(*,1:3)
MAT Newskit(*,1) = Skitmod(*,Screen_x)
MAT Newskit(*,2) = Skitmod(*,Screen_y)
MAT Newskit(*,3) = Skitmod(*,4)
MAT Newskit(37:37,*) = Pen1
MAT Newskit(68:68,*) = Pen2
MAT Newskit(99:99,*) = Pen3
OUTPDT @Path1; Newskit(*)
GOTO Top

Movie:
ASSIGN @Path1 TO *
ASSIGN @Path2 TO *
CALL Display_movie(N, Skitwork$)
IF File_flag = 1 THEN GOTO Finished

Save_file:
DISP "DO YOU WANT TO SAVE THE WORK FILE Y OR N";
LINPUT Ans$
IF Ans$ <> "Y" AND Ans$ <> "N" THEN GOTO 19170
IF Ans$ = "N" THEN GOTO Delete
IF Ans$ = "Y" THEN
DISP "NAME OF FILE TO SAVE UNDER";
LINPUT Name$
WHERE$ = Name$
RENAME Skitwork$ TO Where$
DISP "FILE SAVED UNDER"; Where$
GOTO Finished
END IF

Delete:
PURGE Skitwork$
GOTO Finished

System_isr:
SELECT Twirl$
CASE "ROTATE Y"
CALL Rotate_y(Skitter(*), Skitmod(*), Total(*), Trans(*), Temp(*), Sys_rot_y, Theta, Printflag$)
CASE "ROTATE X"
CALL Rotate_x(Skitter(*), Skitmod(*), Total(*), Trans(*), Temp(*), Sys_rot_x, Theta, Printflag$)
CASE "ROTATE Z"
Theta = -Theta
CALL Rotate_z(Skitter(*), Skitmod(*), Total(*), Trans(*), Temp(*), Sys_rot_z, Theta, Printflag$)
CASE "TRANSLATE"
CALL Translate_3d(Skitter(*), Skitmod(*), Total(*), Trans(*), Temp(*), Tempa(*), Sys_trans_x, Sys_trans_y, Sys_trans_z, Way, Theta, Printflag$)
CASE "PIVOT"
IF Leg_flag$ = "FEMUR A" THEN
Theta = -Theta
Pivotx=(Skitter(96,1)+Skitter(127,1))/2
Pivoy=(Skitter(96,2)+Skitter(127,2))/2
Pivotz=(Skitter(96,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(65,1))^2+(Pivoy-Skitter(65,2))^2+(Pivotz-Skitter(65,3))^2)

CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoy,-Pivotz)

Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Theta,Prop
CALL Trans_fem_orig(Skitter(*),Skitmod(*),Pivotx,Pivoy,Pivotz)
MAT Skitter= Skitmod
END IF

IF Leg_flag$="FEMUR B" THEN
Pivotx=(Skitter(65,1)+Skitter(127,1))/2
Pivoy=(Skitter(65,2)+Skitter(127,2))/2
Pivotz=(Skitter(65,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(96,1))^2+(Pivoy-Skitter(96,2))^2+(Pivotz-Skitter(96,3))^2)

CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoy,-Pivotz)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),60)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Theta,Prop
CALL Trans_fem_orig(Skitter(*),Skitmod(*),Pivotx,Pivoy,Pivotz)
MAT Skitter= Skitmod
END IF

IF Leg_flag$="FEMUR C" THEN
Pivotx=(Skitter(65,1)+Skitter(96,1))/2
Pivoy=(Skitter(65,2)+Skitter(96,2))/2
Pivotz=(Skitter(65,3)+Skitter(96,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(129,1))^2+(Pivoy-Skitter(129,2))^2+(Pivotz-Skitter(129,3))^2)

CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoy,-Pivotz)
CALL Rotate_leg_y(Skitmod(*),Skitmod(*),-60)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,Theta,Prop
CALL Trans_fem_orig(Skitter(*),Skitmod(*),Pivotx,Pivoy,Pivotz)
MAT Skitter= Skitmod
END IF

END SELECT
RETURN

SELECT Leg$
CASE "FEMUR A"
MAT Femur= Skitter(37:67,*)
CALL Trans_fem_orig(Femur(*),Femurmod(*),-Femur(3,1),-Femur(3,2),-Femur(3,3))
CALL Rotate_leg_y(Skitmod(*), Skitmod(*), -60)
CALL Trans_fem_orlg(Skitmod(*), Skitmod(*), Pivotx, Pivoty, Pivotz)

MAT Skitter = Skitmod
MAT Tempa(*, 1:3) = Skitter(*, 1:3)
MAT Temp = Tempa*Total
MAT Skitmod(*, 1:3) = Temp(*, 1:3)

CASE "FEMUR C"
MAT Femur = Skitter(99:129, *)
CALL Trans_fem_orlg(Femur(*), Femurmod(*), -Femur(3, 1, 3))
CALL Rotate_leg_y(Femurmod(*), Femurmod(*), -60)
CALL Trans_fem_orlg(Femurmod(*), Femurmod(*), Pivotx, Pivoty, Pivotz)

MAT Skitter = Skitmod
MAT Tempa(*, 1:3) = Skitter(*, 1:3)
MAT Temp = Tempa*Total
MAT Skitmod(*, 1:3) = Temp(*, 1:3)

IF Freeleg$ = "FREE" THEN
MAT Femurtemp(*, 1:3) = Femurmod(*, 1:3)
MAT Femur = Femurtemp*Total
MAT Skitmod(99:129, 1:3) = Femur(*, 1:3)
GOTO End_leg
END IF

Pivotx = (Skitter(65, 1) + Skitter(96, 1))/2
Pivoty = (Skitter(65, 2) + Skitter(96, 2))/2
Pivotz = (Skitter(65, 3) + Skitter(96, 3))/2
Dist_piv_foot = SQR((Pivotx - Skitter(127, 1))^2 + (Pivoty - Skitter(127, 2))^2 + (Pivotz - Skitter(127, 2))^2)

Dist_y = Pivoty - Skitter(127, 2)
Rot_ang = ASN(Dist_y / Dist_piv_foot)
CALL Trans_fem_orlg(Skitter(*), Skitmod(*), -Pivotx, -Pivoty, -Pivotz)

CALL Rotate_leg_y(Skitmod(*), Skitmod(*), -60)
CALL Trans_fem_orlg(Skitmod(*), Skitmod(*), Pivotx, Pivoty, Pivotz)

MAT Skitter = Skitmod
MAT Tempa(*, 1:3) = Skitter(*, 1:3)
MAT Temp = Tempa*Total
MAT Skitmod(*, 1:3) = Temp(*, 1:3)
CASE "TIBIA A"
MAT Tibia= Skitter(58:67,*)
CALL Trans_fem_orig(Tibia(*),Tibiamod(*),-Tibia(2,1),
Flag=1
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),180)
CALL Rotate_leg_z(Tibiamod(*),Tibiamod(*),Leg$,Theta,
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),180)
CALL Trans_fem_orig(Tibiamod(*),Tibiamod(*),Tibia(2,1)
MAT Skitter(58:67,*)= Tibiamod
IF Freeleg$="FREE" THEN
MAT Tibiatemp(*,1:3)= Tibiamod(*,1:3)
MAT Tibia= Tibiatemp*Total
MAT Skitmod(58:67,1:3)= Tibia(*,1:3)
GOTO End_leg
END IF

Pivotx=(Skitter(96,1)+Skitter(127,1))/2
Pivoty=(Skitter(96,2)+Skitter(127,2))/2
Pivotz=(Skitter(96,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(65,1))^2+(Pivoty-Skitter(65,2))^2+(Pivotz-Skitter(65,3))^2)
Dist_y=Pivoty-Skitter(65,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,-Pivotz)
Flag=0
CALL Rotate_leg_z(Skitmod(*),Skitmod(*),Leg$,-Rot_ang
CALL Trans_fem_orig(Skitmod(*),Skitmod(*),Pivotx,Pivoty,Pivotz)
MAT Skitter= Skitmod
MAT Tempa(*,1:3)= Skitter(*,1:3)
MAT Temp$= Tempa*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)
CASE "TIBIA B"
MAT Tibia= Skitter(89:98,*
CALL Trans_fem_orig(Tibia(*),Tibiamod(*),-Tibia(2,1),
Flag=1
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),60)
CALL Rotate_leg_z(Tibiamod(*),Tibiamod(*),Leg$,Theta,
CALL Rotate_leg_y(Tibiamod(*),Tibiamod(*),-60)
CALL Trans_fem_orig(Tibiamod(*),Tibiamod(*),Tibia(2,1)
MAT Skitter(89:98,*)= Tibiamod
IF Freeleg$="FREE" THEN
MAT Tibiatemp(*,1:3)= Tibiamod(*,1:3)
MAT Tibia= Tibiatemp*Total
MAT Skitmod(89:98,1:3)= Tibia(*,1:3)
GOTO End_leg
END IF

Pivotx=(Skitter(65,1)+Skitter(127,1))/2
Pivoty=(Skitter(65,2)+Skitter(127,2))/2
Pivotz=(Skitter(65,3)+Skitter(127,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(96,1))^2+(Pivoty-Skitter(96,2))^2+(Pivotz-Skitter(96,3))^2)

Dist_y=Pivoty-Skitter(96,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,-Pivotz)

CALL Rotate_leg_y(Skitter(*),Skitmod(*),60)
Flag=0
CALL Rotate_leg_z(Skitter(*),Skitmod(*),Leg$,Rot_ang,
Printflag$,Tib_b_ang,Flag)

CALL Rotate_leg_y(Skitter(*),Skitmod(*),-60)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),Pivotx,Pivoty,Pivotz)

MAT Skitter= Skitmod
MAT Temple(*,1:3)= Skitter(*,1:3)
MAT Temp= Temple*Total
MAT Skitmod(*,1:3)= Temp(*,1:3)

CASE "TIBIA C"
MAT Tibia= Skitter(120:129,*)
CALL Trans_fem_orig(Tibia(*),Tiblamod(*),-Tibia(2,1),
-Tibia(2,2),-Tibia(2,3))
CALL Rotate_leg_y(Tiblamod(*),Tiblamod(*),-60)
Flag=1
CALL Rotate_leg_z(Tiblamod(*),Tiblamod(*),Leg$,Theta,
Printflag$,Tib_c_ang,Flag)

CALL Rotate_leg_y(Tiblamod(*),Tiblamod(*),60)
CALL Trans_fem_orig(Tiblamod(*),Tiblamod(*),Tibia(2,1)
),Tibia(2,2),Tibia(2,3))

MAT Skitter(120:129,*)= Tiblamod

IF Freeleg$="FREE" THEN
MAT Tibiatemp(*,1:3)= Tibiamod(*,1:3)
MAT Tibia= Tibiatemp*Total
MAT Skitmod(120:129,1:3)= Tibia(*,1:3)
GOTO End_leg
END IF

Pivotx=(Skitter(65,1)+Skitter(127,1))/2
Pivoty=(Skitter(65,2)+Skitter(96,2))/2
Pivotz=(Skitter(65,3)+Skitter(96,3))/2
Dist_piv_foot=SQR((Pivotx-Skitter(127,1))^2+(Pivoty-Skitter(127,2))^2+(Pivotz-Skitter(127,3))^2)

Dist_y=Pivoty-Skitter(127,2)
Rot_ang=ASN(Dist_y/Dist_piv_foot)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),-Pivotx,-Pivoty,-Pivotz)

CALL Rotate_leg_y(Skitter(*),Skitmod(*),-60)
Flag=0
CALL Rotate_leg_z(Skitter(*),Skitmod(*),Leg$,Rot_ang,
Printflag$,Tib_c_ang,Flag)

CALL Rotate_leg_y(Skitter(*),Skitmod(*),60)
CALL Trans_fem_orig(Skitter(*),Skitmod(*),Pivotx,Pivoty,Pivotz)
MAT Skitter = Skitmod
MAT Tempa(*,1:3) = Skitter(*,1:3)
MAT Temp = Tempa*Total
MAT Skitmod(*,1:3) = Temp(*,1:3)

21960
21970
21980
21990
22000 1
22010 End_leg:
22020    END SELECT
22030    RETURN
22040 1
22050 1
22060 Windows:
22070    CALL Zoom_pan(Window$,Screenx_win_min,Screenx_win_max,Screeny_win_min,Screeny_win_max,Skitmod(*),Newskit(*),Screen_x,Screen_y)
22080    GOTO Menu
22090 1

22110 Init:

22130    DEG
22140    GINIT
22150    GRAPHICS ON
22160    PLOTTER IS CRT,"INTERNAL"
22170 1
22180    Dump_device=9
22190 1
22200    Plot_device=705
22210 1
22220    READ Skitter(*)
22230 1
22240    MAT Skitmod= Skitter
22250 1
22260    MAT Femurtemp= (1)
22270 1
22280    MAT Tibiatemp= (1)
22290    MAT Trans= IDN
22300 1
22310    MAT Tempo= (1)
22320 1
22330    MAT Total= IDN
22340 1
22350    Menu$="MAIN"
22360 1
22370    Twirl$="ROTATE Y"
22380 1
22390    Freeleq$="FIXED"
22400 1
22410    Way=1
22420 1
22430    Printflag$="OFF"
22440 1
22450    Increment=5
22460 1
22470    Rot_increment=3
22480 1
22490    Screen_x=1
22500    Screen_y=2
22510 1
22520    Sys_trans_x=0
22530    Sys_trans_y=0
22540    Sys_trans_z=0

! INITIALIZE SCREEN, SKITTER

! SET TO DEGREES
! INITIATE GRAPHICS
! TURN G-PLANE ON
! INIT PLOTTER

! READ SKITTER DATA

! INIT TRANS MATRIX TO IDN

! INIT MAIN MENU

! AXIS OF TRANS X=1, Y=2, Z=3

! TRANS INC.
22550 Sys_rot_x = 0
22560 Sys_rot_y = 0
22570 Sys_rot_z = 0
22580!
22590 Fem_a_ang = 0
22600 Fem_b_ang = 0
22610 Fem_c_ang = 0
22620 Tib_a_ang = 90
22630 Tib_b_ang = 90
22640 Tib_c_ang = 90
22650!
22660 Screenx_win_max = 40
22670 Screenx_win_min = 20
22680 Screeny_win_max = 40
22690 Screeny_win_min = 20
22700 SHOW Screenx_win_min, Screenx_win_max, Screeny_win_min, Screeny_win_max
22710!
22720 RETURN
22730!*******************************************************************************
22740!********************************************************************************
22750 Finished:
22760!*******************************************************************************
22770!
22780 Finished: 
22790 GCLEAR
22800 CLEAR SCREEN
22810 SUBEND
22820!
22830 SUB Display_movie(N, Skitwork$)
22840!
22850 OPTION BASE 1
22860 DIM Newskit(129, 3)
22870 ON KBD GOTO Leave
22880 DISP "HIT ANY KEY TO QUIT"
22890 ASSIGN @Path1 TO Skitwork$
22900 ON END @Path1 GOTO Start_again
22910 ENTER @Path1; Newskit(*)
22920 CLEAR SCREEN
22930 GCLEAR
22940 FRAME
22950 AREA INTENSITY 0,0,.2
22960 MOVE -100,-.5
22970 RECTANGLE 200,.5, FILL, EDGE
22980 PLOT Newskit(*)
22990 LINE TYPE 1
23000 GOTO 22910
23010 Start_again:!
23020 ASSIGN @Path1 TO *
23030 GOTO 22890
23040 Leave:!
23050 ASSIGN @Path1 TO *
23060 SUBEND
Appendix C

SKITTER Dynamic Simulation

Program Listing
PARTICULAR ACTUATOR WILL PROVIDE
SUFFICIENT TORQUE AND ANGULAR
VELOCITY TO HAVE SKITTER JUMP A

TO DO THIS, A INVERSE SLIDER CRANK
MECHANISM WILL BE SIMULATED. JOINT
ANGLES, VELOCITIES, AND ACCELERATIONS
WILL BE CALCULATED ALONG WITH THE TORQUES.

PROGRAM WRITTEN BY:
BRICE MACLAREN
GARY MCMURLAY

************************************************************

DIM A(4), B(4), C(4), D(4), E(4), F(4), Rot1(4,4), Rot2(4,4), Rot3(4,4)
DIM Femur(4), Foot(4), Tibcon(4), Oflag(6), Trans1(4,4), Matrix(4,4)
DIM Trans2(4,4), Trans3(4,4), Newfem(4), Newfoot(4), Temp(4,4), Dist(4)
DIM Newb(4), Newd(4), Newe(4), Newfem2(4), Temp2(4,4), Temp3(4)
DIM Ftorque(800), Ttorque(800), Fomega(800), Tomega(800), Fhp(800), Thp(200)
DIM Origb(4), Orlgd(4), Orlige(4), Newfoot2(4)
DIM Temp$[8]

REDMETHODS

CALL Invar(Mfemur, Ifemur, Flen, Mtibla, Itibia, Tlen)
MAT Trans3= IDN
MAT Trans2= IDN
MAT Trans1= IDN

CALL Invar(Mfemur, Ifemur, Flen, Mtibla, Itibia, Tlen)
MAT Trans3= IDN
MAT Trans2= IDN
MAT Trans1= IDN

Angf=23.62
Angt=27.5
Aforce=100
Avel=5
Avel=Avel/12
Atorque=100
Aomega=2

What follows are the original point locations for the various
REM CONNECT POINTS -- THEY ARE, IN ORDER, PNT.A,B,C,D,E, & F. PLEASE REM THE DOCUMENTATION FOR DEFINITIONS OF THESE POINTS.

REM*****************************************************************************
DATA -1.95,7.5095,0,1
DATA 9.576,4.09,0,1
DATA 0,0,0,1
DATA 11.411,-4.09,0,1
DATA 23.714,-11.7685,0,1
REM*****************************************************************************
DATA 20,0,0,1
DATA -20,0,0,1
DATA 6.95,-11.71,0,1
REM*****************************************************************************
READ A(*)
READ B(*)
READ C(*)
READ D(*)
READ E(*)
REM*****************************************************************************
DATA 1.95,7.5095,0,1
DATA 9.576,4.09,0,1
DATA 0,0,0,1
DATA 11.411,-4.09,0,1
DATA 23.714,-11.7685,0,1
REM*****************************************************************************
READ Femur(*)
READ Foot(*)
READ Tibcon(*)
REM CONVERT INCHES TO FEET
FOR I=1 TO 3
  A(I)=A(I)/12
  B(I)=B(I)/12
  C(I)=C(I)/12
  D(I)=D(I)/12
  E(I)=E(I)/12
  Femur(I)=Femur(I)/12
  Foot(I)=Foot(I)/12
  Tibcon(I)=Tibcon(I)/12
NEXT I
REM*****************************************************************************
Flag=0
Simflag=0
Rflag=1
Oflag=0
FOR I=1 TO 6
  IF Oflag(I)=0 THEN
    CLEAR SCREEN
    CALL Invar(Mfemur,Ifemur,Flen,Mtibla,Itibia,Tlen)
    IF Flag<>0 THEN
      CALL Printvar(Jdist,Adist,Wgt,G,Mfemur,Flen,Mtibla,Tlen,Angf,Angt,A(*),B(*),C(*),D(*),E(*),Beta,Iota,Aforce,Avel,Actflag,Atorque,Aomega)
    ELSE
      DISP "CHOOSE TYPE OF ACTUATOR"
    END IF
  END IF
REM*****************************************************************************
SELECT Flag
CASE 0
ON KEY 0 LABEL "ROTARY ACT." GOTO Ract
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "" GOTO Try
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "LINEAR ACT." GOTO Lact
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "STOP" GOTO St
ON KEY 8 LABEL "" GOTO Try
ON KEY 9 LABEL "" GOTO Try
CASE 1
ON KEY 0 LABEL "JUMP DISTANCE" GOTO Jd
ON KEY 1 LABEL "ACCELERATION DIST" GOTO Ad
ON KEY 2 LABEL "LEG PROPERTIES" GOTO Lp
ON KEY 3 LABEL "JOINT LOCATIONS" GOTO Jnt
ON KEY 4 LABEL "ACTUATORS" GOTO Act
ON KEY 5 LABEL "CHANGE ACT" GOTO Cha
ON KEY 6 LABEL "DATA FILES" GOTO Fn
ON KEY 7 LABEL "STOP" GOTO St
ON KEY 8 LABEL "PRINTER IS ?" GOTO Prnt
ON KEY 9 LABEL "RUN DATA" GOTO Rn
CASE 2
ON KEY 0 LABEL "FEMUR MASS" GOTO Fm
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "INIT FEMUR ANGLE" GOTO Fang
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "LEG LENGTH" GOTO Llen
ON KEY 5 LABEL "TIBIA MASS" GOTO Tm
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "INIT TIBIA ANGLE" GOTO Tang
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn
CASE 3
ON KEY 0 LABEL "POINT A" GOTO Ba1
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "POINT B" GOTO Ba2
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "POINT D" GOTO Tal
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "POINT E" GOTO Ta2
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn
CASE 4
ON KEY 0 LABEL "FEMUR FILES" GOTO Cf
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "TIBIA FILES" GOTO Tibfiles
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "SIMULATION" GOTO Simfil
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "CLOSE FILES" GOTO Clf
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn
CASE 5
ON KEY 0 LABEL "TORQ VS JDIST" GOTO Ftj
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "OMEGA VS JDIST" GOTO Foj
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "HP VS JDIST" GOTO Fhj
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn

CASE 6
ON KEY 0 LABEL "TORQ VS JDIST" GOTO Ttj
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "OMEGA VS JDIST" GOTO Toj
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "HP VS JDIST" GOTO Thj
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn

CASE 7
ON KEY 0 LABEL "MAX FORCE" GOTO Actf
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "MAX VELOCITY" GOTO Actv
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "" GOTO Try
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn

CASE 8
ON KEY 0 LABEL "MAX TORQUE" GOTO Actt
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "MAX OMEGA" GOTO Acto
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "" GOTO Try
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn

CASE 9
ON KEY 0 LABEL "CRT" GOTO Pcrf
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "" GOTO Try
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "LASER" GOTO Las
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn

CASE 10
ON KEY 0 LABEL "JUMP DISTANCE" GOTO Jd
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "" GOTO Try
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "ACCELERATION" GOTO Ad
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn
CASE 11
ON KEY 0 LABEL "FIND MAX VALS" GOTO Fmv
ON KEY 1 LABEL "" GOTO Try
ON KEY 2 LABEL "" GOTO Try
ON KEY 3 LABEL "" GOTO Try
ON KEY 4 LABEL "INCREMENTAL" GOTO Inc
ON KEY 5 LABEL "" GOTO Try
ON KEY 6 LABEL "" GOTO Try
ON KEY 7 LABEL "" GOTO Try
ON KEY 8 LABEL "MAIN MENU" GOTO Mm
ON KEY 9 LABEL "RUN DATA" GOTO Rn
END SELECT
GOTO 2510

BEGIN SOFTKEY DEFINATIONS

Ract:
Actflag=1
Flag=1
GOTO Menu

Lact:
Actflag=2
Flag=1
GOTO Menu

Rcond:
Flag=11
GOTO Menu

Jd:
Disp "DISTANCE YOU DESIRE SKITTER TO JUMP (IN INCHES)"
INPUT Jdist
Jdist=Jdist/12
GOTO Menu

Ad:
Disp "ACCELERATION DISTANCE FOR FOOT (IN INCHES)"
INPUT Adist
Adist=Adist/12
GOTO Menu

Lp:
Flag=2
GOTO Menu

Jnt:
Flag=3
GOTO Menu

Act:
IF Actflag=1 THEN
Flag=8
ELSE
Flag=7
END IF
GOTO Menu

Cha:
Flag=0
GOTO Menu

Prnt:
Flag=9
GOTO Menu

Try:
Disp "BAD CHOICE -- TRY AGAIN"
GOTO Menu

Fn:
!
EXIT THE PROGRAM

RUN PROGRAM WITH DATA AS IT IS

INPUT WEIGHT OF FEMUR

INPUT LENGTH OF LEGS

INPUT WEIGHT OF TIBIA

INPUT INITIAL ANGLE OF FEMUR

INPUT INITIAL ANGLE BETWEEN FEMUR AND TIBIA

GO BACK TO MAIN MENU

CHANGE PNT. A

POINT A - X COORDINATE (DEFAULT = ",A(1)*12,"IN.)

POINT A - Y COORDINATE (DEFAULT = ",A(2)*12,"IN.)

POINT A - Z COORDINATE (DEFAULT = ",A(3)*12,"IN.)
ELSE
A(3)=VAL(Temp$)
A(3)=A(3)/12
END IF
GOTO Menu

Ba2:  ! CHANGE PNT. B
DISP "POINT B - X COORDINATE (DEFAULT = ",B(1)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
B(1)=VAL(Temp$)
B(1)=B(1)/12
END IF
DISP "POINT B - Y COORDINATE (DEFAULT = ",B(2)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
B(2)=VAL(Temp$)
B(2)=B(2)/12
END IF
DISP "POINT B - Z COORDINATE (DEFAULT = ",B(3)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
B(3)=VAL(Temp$)
B(3)=B(3)/12
END IF
GOTO Menu

Ta1:  ! CHANGE PNT. D
DISP "POINT D - X COORDINATE (DEFAULT = ",D(1)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
D(1)=VAL(Temp$)
D(1)=D(1)/12
END IF
DISP "POINT D - Y COORDINATE (DEFAULT = ",D(2)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
D(2)=VAL(Temp$)
D(2)=D(2)/12
END IF
DISP "POINT D - Z COORDINATE (DEFAULT = ",D(3)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
D(3)=VAL(Temp$)
D(3)=D(3)/12
END IF
GOTO Menu

Ta2:  ! CHANGE PNT. E
DISP "POINT E - X COORDINATE (DEFAULT = ",E(1)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
E(1)=VAL(Temp$)
E(1)=E(1)/12
END IF
DISP "POINT E - Y COORDINATE (DEFAULT = ",E(2)*12,"IN.)";
LINPUT Temp$
IF Temp$="" THEN
ELSE
  z(2)=VAL(Temp$)
  E(2)=E(2)/12
END IF
DISP "POINT E - Z COORDINATE (DEFAULT = ",E(3)*12,"IN.)";
LINPUT Temp$
IF Temp$="" TI'EN
ELSE
  E(3)=VAL(Temp$)
  E(3)=E(3)/12
END IF
CREATE DATA FILES FOR FEMUR
CREATE DATA FILES FOR TIBIA
CLOSE ALL DATA FILES

 This creates a BDAT file the output of the angles for the simulation of skitter

DISP "FILE NAME FOR THE SIMULATION FILE ";
LINPUT Sim$
CREATE BDAT Sim$,200
ASSIGN @Pathsim TO Sim$
Simflag=1
GOTO Menu
CREATE TORQUE vs JUMP DISTANCE FILE FOR FEMUR

DISP "FILE NAME FOR TORQUE VS JUMP DISTANCE ";
LINPUT Tvj$
CREATE ASCII Tvj$,25
ASSIGN @Path1 TO Tvj$
Oflag(1)=1
GOTO Menu
CREATE OMEGA vs JUMP DISTANCE FILE FOR FEMUR

DISP "FILE NAME FOR OMEGA VS JUMP DISTANCE ";
LINPUT Ovj$
CREATE ASCII Ovj$,25
ASSIGN @Path2 TO Ovj$
Oflag(2)=1
GOTO Menu
CREATE HP vs JUMP DISTANCE FILE FOR FEMUR

DISP "FILE NAME FOR HORSE POWER VS JUMP DISTANCE ";
LINPUT Hvj$
CREATE ASCII Hvj$,25
ASSIGN @Path3 TO Hvj$
Oflag(3)=1
GOTO Menu

CREATE TORQUE vs JUMP DISTANCE FILE FOR TIBIA

4810 DISP "FILE NAME FOR TORQUE VS JUMP DISTANCE ";
4820 LINPUT Tvj$
4830 CREATE ASCII Tvj$,25
4840 Oflag(4)=1
4860 GOTO Menu

CREATE OMEGA vs JUMP DISTANCE FILE FOR TIBIA

4870 TOj: 
4880 DISP "FILE NAME FOR OMEGA VS JUMP DISTANCE ";
4890 LINPUT Ovj$
4900 CREATE ASCII Ovj$,25
4910 Oflag(5)=1
4930 GOTO Menu

CREATE HP vs JUMP DISTANCE FILE FOR TIBIA

4940 Thj:
4950 DISP "FILE NAME FOR HORSE POWER VS JUMP DISTANCE ";
4960 LINPUT Hvj$
4970 CREATE ASCII Hvj$,25
4980 Oflag(6)=1
4990 GOTO Menu

INPUT MAXIMUM FORCE ACTUATOR CAN EXERT

5010 Actf:
5020 DISP "MAXIMUM FORCE ACTUATOR CAN EXERT (IN POUNDS FORCE) ";
5030 INPUT Aforce
5040 GOTO Menu

INPUT MAXIMUM VELOCITY ACTUATOR CAN ACHIEVE

5050 Actv:
5060 DISP "MAXIMUM ACTUATOR VELOCITY ACHIEVE (IN IN/SEC) ";
5070 INPUT Avel
5080 Avel=Avel/12
5090 GOTO Menu

INPUT MAXIMUM TORQUE ACTUATOR CAN EXERT

5100 Actt:
5110 DISP "MAXIMUM TORQUE ACTUATOR CAN EXERT (IN FT-LBS) ";
5120 INPUT Atorque
5130 GOTO Menu

INPUT MAXIMUM VELOCITY ACTUATOR CAN ACHIEVE

5140 Acto:
5150 DISP "MAXIMUM ACTUATOR OMEGA ACHIEVE (IN RAD/SEC) ";
5160 INPUT Aomega
5170 GOTO Menu

MAKE PRINTER CRT

5180 Pcrt:
5190 PRINTER IS CRT
5200 Flag=1
5210 GOTO Menu

MAKE PRINTER THE LASER JET

5220 Las:
5230 PRINTER IS 9
5240 Flag=1
5250 GOTO Menu

RUNS THE PROGRAM TO FIND MAX TORQUE & OMEGA

5260 Fmv:
5270 DISP "NUMBER OF STEPS PER INCH OF ACCELERATION ";
5280 INPUT Delstep
5290 Delstep=1/Delstep
5300 Rflag=2
5310 Flag=1
5320 GOTO Rest

RUNS THE PROGRAM ON AN INCREMENTAL SETTING

5330 Inc:
5340 Rflag=1
5350 Flag=1
5360 GOTO Rest

RUN THE PROGRAM WITH THE VALUES AS SET

5370 Rest:
5380 Flag=1
CLEAR SCREEN

400  IF NO FILE ASSIGNMENTS HAVE BEEN MADE, THEN OUTPUT IS TO THE CRT
410  
420  IF Oflag(1)=0 THEN
430       ASSIGN @Path1 TO CRT
440  END IF
450  IF Oflag(2)=0 THEN
460       ASSIGN @Path2 TO CRT
470  END IF
480  IF Oflag(3)=0 THEN
490       ASSIGN @Path3 TO CRT
500  END IF
510  IF Oflag(4)=0 THEN
520       ASSIGN @Path4 TO CRT
530  END IF
540  IF Oflag(5)=0 THEN
550       ASSIGN @Path5 TO CRT
560  END IF
570  IF Oflag(6)=0 THEN
580       ASSIGN @Path6 TO CRT
590  END IF
600  !*******************************************************************
610  |** START THE PROGRAM **|
620  |
630  |*******************************************************************
640  |
650  |*******************************************************************
660  |
670  |
680  | DEFINE YOUR END CONDITIONS
690  
700  Maxvel=SQR(2*G*Jdist)
710  Totaltime=ABS(2*Adist/Maxvel)
720  Accel=Maxvel/Totaltime
730  |
740  | DETERMINE ANGLES FOR THIS POSITION OF THE FOOT
750  |
760  CALL Findfoot(-l*Beta,Iota,Femur(*),Foot(*),Newfoot(*),Newfem(*))
770  |
780  |
790  | DEFINE ORIGINAL LOCATION OF ACTUATOR CONNECT POINTS FOR THIS
800  | CONFIGURATION OF THE LEGS
810  |
820  CALL Rot(-l*Beta,Trans1(*))
830  MAT Origb= Trans1*B
840  MAT Origd= Trans1*D
850  MAT Temp2= E-Femur
860  Temp2(4)=1
870  MAT Templ= Trans3*Temp2
880  Xnew=Newfoot(1)
890  Ynew=Newfoot(2)
900  Lac=SQR(Xnew^2+(Ynew)^2)
910  CALL Sss(Tlen,Lac,Flen,A3)
920  A3=A3-PI/2
930  CALL Rot(A3,Trans2(*))
940  CALL Trans(Newfem(*),Trans2(*))
950  MAT Orige= Trans2*Templ
960  |
970  | SET END POINTS FOR LOOPING OVER SLIDER CRANK
980  |
990  )
IF Rflag=1 THEN
Delstep=1
Y0=Delstep
ELSE
Y0=Delstep
END IF
Endpnt=Adist*12
Theta2=-1+Beta
Iota2=Iota
!
BEGIN LOOP OVER ACCELERATION DISTANCE
!
FOR Y=Y0 TO Endpnt STEP Delstep
Y1=Y/12
T=ABS(SQR(2*Y1/Accel))
V=Accel*T
!
Determine Joint Angles
!
Xnew=Newfoot(1)
Ynew=Newfoot(2)
Lac=SQR(Xnew^2+(Ynew-(Y1))^2)
Psi=ATN((Ynew-(Y1))/Xnew)
CALL Sss(Flen,Tlen,Lac,A3)
IF Psi<0. THEN A3=-1*A3
Theta=Psi-A3
IF Psi>0. THEN
Gamma=PI/2-Psi
ELSE
Gamma=Psi+PI/2
END IF
CALL Sss(Lac,Flen,Tlen,A3)
IF Gamma>0. THEN
Phi=Gamma-A3
ELSE
Phi=A3-Gamma
END IF
!
NOW OUTPUT ANGLES FOR SIMULATION IF SIMFLAG=1
!
CALL Sss(Tlen,Lac,Flen,Iotal)
Theta1=Theta
Angle1=(Theta1-Theta2)*180/PI
Angle2=(Iotal-Iota2)*180/PI
Free=2.0
OUTPUT @Pathsim;T
OUTPUT @Pathsim;Free
OUTPUT @Pathsim;Angle1
OUTPUT @Pathsim;Angle2
Theta2=Theta1
Iota2=Iotal
!
NOW CALCULATE THE PHYSICAL PARAMETERS
!
Fomega(I)=ABS((V*COS(Phi))/(Flen*(COS(Theta)*COS(Phi)-SIN(Theta)*SIN(Phi))))
Tomega(I)=ABS((Fomega(I)*SIN(Theta))/COS(Phi))
Numbc = -1*Accel*Tlen*SIN(Theta) - Tomega(I)^2*Tlen^2*(COS(Phi)*SIN(Theta) + SIN(Phi)*COS(Theta)) + Fomega(I)^2*Tlen^2*(COS(Theta)^2 - SIN(Theta)^2)
Numbc = -1*Tlen^2*(COS(Phi)*COS(Theta) + SIN(Phi)*SIN(Theta))
Falpha = (Talpha*Tlen*COS(Phi)/2 - Tomega(I)^2*Tlen*SIN(Phi) + Fomega(I)^2*Tlen^2*COS(Phi)/2 - Flen*COS(Phi))/Tlen*SIN(Theta)
Axtibia = -1*Talpha*Tlen*COS(Phi)/2 + Tomega(I)^2*Tlen*SIN(Phi)/2
Aytibia = -1*Talpha*SIN(Phi)*Tlen/2 - Tomega(I)^2*Tlen*COS(Phi)/2
Vbx = Mtxbia*Axtibia
Fby = Wgt/3 - Mtibia*Aytibia - Mtbia*G
Ftorque(I) = ABS(I*Ifemur*Falpha + Fbx*Flen*SIN(Theta) + Mfemur*G*Flen*COS(Theta)/2 - Fby*Flen*COS(Theta))
Ttorque(I) = ABS(-I*Itibia*Talpha - Mtbia*G*Tlen*SIN(Phi)/2 + Wgt*Tlen*SIN(Phi)/2)

IF ACTUATOR IS A LINEAR ONE, THEN CALCULATE THE MOMENT ARM. IF IT IS A ROTARY ONE, THEN COMPARE TORQUE'S AND OMEGA'S NEEDED WITH THE ONES YOU CAN SUPPLY.

Cnt = 0 ! CNT COUNTS THE NUMBER OF LINES PRINTED ON THE SCREEN SO THAT SCROLLING OF THE SCREEN CAN BE PREVENTED.

NOW, CALCULATE THE MOMENT ARM FOR THE LINEAR ACTUATOR.

MAT Trans3 = IDN
MAT Trans2 = IDN
CALL Rot(Theta, Trans3(*))
MAT Newfem = Trans3*Femur
MAT Newb = Trans3*B
MAT Newd = Trans3*D
MAT Temp2 = E-Femur
Temp2(4) = 1
MAT Templ = Trans3*Temp2
CALL Sss(Tlen, Lac, Flen, A3)
A3 = A3 - PI/2
CALL Rot(A3, Trans2(*))
CALL Trans(Newfem(*), Trans2(*))
MAT Newe = Trans2*Templ
CALL Eqline(A(*), Newb(*), C(*), Farm)
CALL Eqline(Newd(*), Newe(*), Newfem(*), Tarm)
A3 = A3 + PI/2
CALL Rot(A3, Trans2(*))
MAT Temp1 = Trans3*Foot
MAT Newfoot2 = Trans2*Temp1

CAN GIVEN ACTUATOR SUPPLY POWER?

Test = ABS(Ftorque(I)/Farm)
Test2 = ABS(Fomega(I)*Farm)
Testf = ABS(Ttorque(I)/Tarm)
Testv = ABS(Tomega(I)*Tarm)
IF Test > Aforce THEN
IF Rflag = 1 THEN
PRINT CHR$(129)
PRINT "FEMUR ACTUATOR IS NOT GOOD ENOUGH - FORCE NOT ENOUGH!!!"
PRINT CHR$(128)
MINARM = Ftorque(I)/Aforce
PRINT USING "29A,DDDDD.DD"; "MINIMUM MOMENT ARM NEEDED IS "; MINARM*1
PRINT USING "29A,DDD.DD";"PRESENTLY, HAVE MOMENT ARM = ";Farm*12;" IN."

PRINT USING "30A,DDDDDD.DD,17A,DDDD.DD";"OR INCREASE ACTUATOR FORCE TO ";Testf;" LBF. INSTEAD OF ";Aforce;" LBF

PRINT
Cnt=Cnt+7
ELSE
Femfflag=2
END IF

IF Test2>Avel THEN
IF Rflag=1 THEN
PRINT CHR$(129)
PRINT "FEMUR ACTUATOR IS NOT GOOD ENOUGH - VELOCITY NOT ENOUGH!!"
Maxarm=Avel/Fomega(I)
PRINT USING "49A,DDD.DD";"MAXIMUM MOMENT ARM POSSIBLE FOR THIS ACTUATOR IS ";Maxarm*12;" IN."
PRINT USING "29A,DDD.DD";"PRESENTLY, HAVE MOMENT ARM ";Farm*12;" IN."
PRINT USING "33A,DDD.DD,19A,DDDD.DD";"OR INCREASE ACTUATOR VELOCITY TO ";Test2*12;" IN/SEC INSTEAD OF ";Avel*12;" IN/SEC

PRINT
Cnt=Cnt+7
ELSE
Femvflag=2
END IF

END IF

IF Cnt>9 THEN
DISP "HIT ANY KEY TO CONTINUE";
ON KBD GOTO 7630
GOTO 7620
Cnt=0
CLEAR SCREEN
END IF

IF Testf>Aforce THEN
IF Rflag=1 THEN
PRINT CHR$(129)
PRINT "TIBIA ACTUATOR IS NOT GOOD ENOUGH - FORCE NOT ENOUGH!!"
Minarm=Ttorque(I)/Aforce
PRINT USING "29A,DDD.DD";"MINIMUM MOMENT ARM NEEDED IS ";Minarm*12;" IN."
PRINT USING "29A,DDD.DD";"PRESENTLY, HAVE MOMENT ARM = ";Tarm*12;" IN."
PRINT USING "30A,DDDDDD.DD,17A,DDDD.DD";"OR INCREASE ACTUATOR FORCE TO ";Testf;" LBF. INSTEAD OF ";Aforce;" LBF

PRINT
Cnt=Cnt+7
ELSE
Tibfflag=2
END IF

END IF

IF Cnt>9 THEN
DISP "HIT ANY KEY TO CONTINUE";
ON KBD GOTO 7630
GOTO 7620
Cnt=0
CLEAR SCREEN
END IF

IF Testv>Avel THEN
  IF Rflag=1 THEN
    PRINT CHR$(129)
    PRINT "TIBIA ACTUATOR IS NOT GOOD ENOUGH - VELOCITY NOT ENOUGH!!!"
    PRINT CHR$(128)
    Maxarm=Avel/Tomega(I)
    PRINT USING "49A, DDD.DD"; "MAXIMUM MOMENT ARM POSSIBLE FOR THIS ACTUATOR IS ";Maxarm*12;" IN."
    PRINT USING "29A, DDD.DD"; "PRESENTLY, HAVE MOMENT ARM = ";Tarm*12;" LBS"
    PRINT USING "33A, DDD.DD,19A, DDD.DD"; "OR INCREASE ACTUATOR VELOCITY > ";Testv*12;" IN/SEC INSTEAD OF ";Avel*12;" IN/SEC"
  PRINT
  Cnt=Cnt+7
  ELSE
    Tibvflag=2
  END IF

END IF

NOW, FOR A ROTARY ACTUATOR

ELSE

IF Ftorque(I)>Atorque THEN
  IF Rflag=1 THEN
    PRINT CHR$(129)
    PRINT "FEMUR ACTUATOR IS NOT GOOD ENOUGH - TORQUE NOT ENOUGH"
    PRINT CHR$(128)
    PRINT USING "15A, DDD.DD,35A, DDD.DD"; "NEED TORQUE OF ";Ftorque(I);" LBS,BUT ACTUATOR ONLY SUPPLIES ";Atorque"
    PRINT
    Cnt=Cnt+5
    ELSE
      Femfflag=2
    END IF

END IF

IF Fomega(I)>Aomega THEN
  IF Rflag=1 THEN
    PRINT CHR$(129)
    PRINT "FEMUR ACTUATOR IS NOT GOOD ENOUGH - OMEGA NOT ENOUGH"
    PRINT CHR$(128)
    PRINT USING "14A, DDD.DD,36A, DDD.DD"; "NEED OMEGA OF ";Fomega(I);" RAD/SEC,BUT ACTUATOR ONLY SUPPLIES ";Aomega"
    PRINT
    Cnt=Cnt+5
    ELSE
      Femtflag=2
    END IF

END IF

IF Ttorque(I)>Atorque THEN
  IF Rflag=1 THEN
    PRINT CHR$(129)
    PRINT "TIBIA ACTUATOR IS NOT GOOD ENOUGH - TORQUE NOT ENOUGH"
    PRINT CHR$(128)
    PRINT USING "15A, DDD.DD,35A, DDD.DD"; "NEED TORQUE OF ";Ttorque(I);" LBS,BUT ACTUATOR ONLY SUPPLIES ";Atorque"
    PRINT
    Cnt=Cnt+5
    ELSE
      Tibfflag=2
    END IF
END IF
8210 IF Cnt>11 THEN
8220 DISP "HIT ANY KEY TO CONTINUE";
8230 ON KBD GOTO 8250
8240 GOTO 8240
8250 Cnt=0
8260 CLEAR SCREEN
8270 END IF
8280 IF Tomega(I)>Aomega THEN
8290 IF Rflag=1 THEN
8300 PRINT CHR$(129)
8310 PRINT "TIBIA ACTUATOR IS NOT GOOD ENOUGH - OMEGA NOT ENOUGH"
8320 PRINT CHR$(128)
8330 PRINT USING "14A,DDD.DD,36A,DDD.DD";"NEED OMEGA OF ";Tomega(I);" RA
8340 PRINT
8350 Cnt=Cnt+5
8360 ELSE
8370 Tibvflag=2
8380 END IF
8390 END IF
8400 END IF
8410 LOOP BACK FOR ACCELERATION DISTANCE
8430 IF Rflag=1 THEN
8450 CALL Outdata(jdist,Adist,Ftorque(*),Ttorque(*),Fomega(*),Tomega(*),Fhp (*),Thp(*),@Path1,@Path2,@Path3,@Path4,@Path5,@Path6,Oflag(*),Y,Cnt,I)
8460 END IF
8470 I=I+1
8480 NEXT Y
8500 WRITE LINE TO END SIMULATION FILE
8520 IF Simflag=1 THEN
8530 Endnum=999.0
8540 OUTPUT @Pathsim;Endnum
8550 OUTPUT @Pathsim;Free
8560 OUTPUT @Pathsim;Free
8570 OUTPUT @Pathsim;Free
8580 END IF
8600 NOW CALCULATE THE STROKE LENGTH FOR THE ACTUATOR
8610 Fsl=ABS(SQR((Origb(1)-A(1))^2+(Origb(2)-A(2))^2)-SQR((Newb(1)-A(1))^2+(New
b(2)-A(2))^2))
8630 Tsl=ABS(SQR((Origd(1)-Orlgd(1))^2+(Origd(2)-Orlgd(2))^2)-SQR((Newd(1)-Newe
(1))^2+(Newd(2)-Newe(2))^2))
8640 CLEAR SCREEN
8660 IF THE MAXIMUM VALUES WERE DESIRED, THEN THIS SECTION
8680 DETERMINES THE MAXIMUM TORQUE AND ANGULAR VELOCITY
8690 IF Actflag=2 THEN
8710 PRINT USING "22A,DD.DD";"FEMUR STROKE LENGTH = ";Fsl*12;" IN."
8720 PRINT USING "22A,DD.DD";"TIBIA STROKE LENGTH = ";Tsl*12;" IN."
8730 END IF
IF Femflag=2 THEN
    PRINT "FEMUR ACTUATOR NOT GOOD ENOUGH!! NEED MORE TORQUE ABOUT JOINT!!"
END IF

IF Femflag=2 THEN
    PRINT "FEMUR ACTUATOR NOT GOOD ENOUGH!! NEED MORE ANGULAR VELOCITY ABOUT JOINT!!"
END IF

IF Tibflag=2 THEN
    PRINT "TIBIA ACTUATOR NOT GOOD ENOUGH!! NEED MORE TORQUE ABOUT JOINT!!"
END IF

IF Tibflag=2 THEN
    PRINT "TIBIA ACTUATOR NOT GOOD ENOUGH!! NEED MORE ANGULAR VELOCITY ABOUT JOINT!!"
END IF

PRINT MAT SEARCH Ftorque,MAX;Maxft
PRINT MAT SEARCH Ttorque,MAX;Maxtt
PRINT MAT SEARCH Fomega,MAX;Maxfo
PRINT MAT SEARCH Tomega,MAX;Maxto
PRINT MAT SEARCH Fhp,MAX;Maxfhp
PRINT MAT SEARCH Thp,MAX;Maxthp
PRINT USING -3SA,DDDD.DDO'"MAXIMUM TORQUE ABOUT THE FEMUR WAS ";Maxft;" FT-LBS"
PRINT USING "45A,DD.DDD";'MAXIMUM ANGULAR VELOCITY ABOUT THE FEMUR WAS ";Maxfo;" RAD/SEC"
PRINT USING "35A,DDDD.DD";'MAXIMUM TORQUE ABOUT THE TIBIA WAS ";Maxtt;" FT-LBS"
PRINT USING "45A,DD.DDD";'MAXIMUM ANGULAR VELOCITY ABOUT THE TIBIA WAS ";Maxto;" RAD/SEC"
PRINT USING "42A,D.DDD";'MAXIMUM HORSE POWER ABOUT THE FEMUR WAS ";Maxthp;" HP"
PRINT USING "42A,D.DDD";'MAXIMUM HORSE POWER ABOUT THE TIBIA WAS ";Maxthp;" HP"

DISP "HIT ANY KEY TO CONTINUE";
ON KBD GOTO 9030
GOTO 9020
GOTO 9010
GOTO 9000
STOP PROGRAM
END

SUB Printvar(Jdist,Adist,Wgt,G,Mfemur,Flen,Mtibia,Tlen,Angf,Angt,A(*),B(*),C(*),D(*),E(*),Beta,Iota,Aforce,Avel,Actflag,Actt,Acto)
  PRINT "HERE ARE THE PRESET PARAMETERS:"
  PRINT "INITIAL ANGLE FEMUR AND HORIZONTAL = ";Beta*180/PI;"DEG"
  PRINT "INITIAL ANGLE FEMUR AND TIBIA = ";Iota*180/PI;"DEG"
  PRINT "JUMP DISTANCE = ";Jdist*12;"IN."
PRINT "ACCELERATION DISTANCE = ";Adist*12;"IN."
PRINT "WEIGHT OF SKITTER = ";Wgt;"LBF"
PRINT "FEMUR'S WEIGHT = ";Mfemur;"G";"LBF"
PRINT "FEMUR'S LENGTH = ";Flen*12;"IN."
PRINT "TIBIA'S WEIGHT = ";Mtibia;"G";"LBF"
PRINT "TIBIA'S LENGTH = ";Tlen*12;"IN."
IF Actflag=2 THEN
PRINT USING "37A,DDD.DD,2X,DDD.DD,2X,DDD.DD";"POINT A COORDINATES ARE 
(IN INCHES): ";A(1)*I2,A(2)*I2,A(3)*I2
PRINT USING "37A,DDD.DD,2X,DDD.DD,2X,DDD.DD";"POINT B COORDINATES ARE 
(IN INCHES): ";B(1)*I2,B(2)*I2,B(3)*I2
PRINT USING "37A,DDD.DD,2X,DDD.DD,2X,DDD.DD";"POINT C COORDINATES ARE 
(IN INCHES): ";C(1)*I2,C(2)*I2,C(3)*I2
PRINT USING "37A,DDD.DD,2X,DDD.DD,2X,DDD.DD";"POINT D COORDINATES ARE 
(IN INCHES): ";D(1)*I2,D(2)*I2,D(3)*I2
PRINT USING "37A,DDD.DD,2X,DDD.DD,2X,DDD.DD";"POINT E COORDINATES ARE 
(IN INCHES): ";E(1)*I2,E(2)*I2,E(3)*I2
END IF
IF Actflag=2 THEN
PRINT USING "17A,DDD.DD";"ACTUATOR FORCE = ";Aforce;" LBF"
PRINT USING "20A,DDD.DD";"ACTUATOR VELOCITY = ";Avel*12;" IN/SEC"
ELSE
PRINT USING "18A,DDD.DD";"ACTUATOR TORQUE = ";Actt;" FT-LBS"
PRINT USING "17A,DDD.DD";"ACTUATOR OMEGA = ";Acto;" RAD/SEC"
END IF
END SUBROUTINE
SUB Invar(Mfemur,Ifemur,Flen,Mtibia,Itibia,Tlen)
IFemur=0.252+Mfemur*(Flen/2)^2
Itibia=0.1495+Mtibia*(Tlen/2)^2
END SUBROUTINE
SUB Outdata(Jdist,Adist,Ftorque(*),Ttorque(*),Fomega(*),Tomega(*),Fhp(*),Thp(*),@Path1,@Path2,@Path3,@Path4,@Path5,@Path6,Oflag(*),Y,Cnt,I)
Temp=0
FOR J=1 TO 6
IF Oflag(J)=0 THEN Temp=1
NEXT J
IF Cnt>13 THEN
DISP "HIT ANY KEY TO CONTINUE";
ON KBD GOTO 9770
GOTO 9760
Cnt=0
!...
CLEAR SCREEN
9790 END IF
9800 IF Temp=1 THEN
9810 PRINT USING "DDD.D,32A":Y/(Adist*12)*100;" % THROUGH ACCELERATION DISTANCE"
9820 PRINT
9830 Cnt=Cnt+3
9850 END IF
9860 IF Cnt>14 THEN
9870 PRINT USING "I6A,DDD.DD";"JUMP DISTANCE = ";Jdist*12;" IN."
9880 PRINT USING "24A,DDD.DD";"ACCELERATION DISTANCE = ";Adist*12;" IN"
9890 Cnt=Cnt+2
9900 END IF
9910 IF Cnt>15 THEN
9920 CLEAR SCREEN
9930 END IF
9940 IF Oflag(1)=1 THEN
9950 OUTPUT @Path1;Jdist*12,Ftorque(I)
9960 END IF
9970 IF Cnt>15 THEN
9980 CLEAR SCREEN
9990 END IF
10000 ELSE
10010 IF Cnt>15 THEN
10020 PRINT USING "16A,DDD.DD";"JUMP DISTANCE = ";Jdist*12;" IN."
10030 PRINT USING "24A,DDD.DD";"ACCELERATION DISTANCE = ";Adist*12;" IN"
10040 Cnt=Cnt+2
10050 END IF
10060 IF Cnt>15 THEN
10070 PRINT USING "15A,DDDD.DD";"FEMUR TORQUE = ";Ftorque(I);" FT-LB"
10080 END IF
10090 Cnt=Cnt+1
10100 END IF
10110 IF Oflag(2)=1 THEN
10120 OUTPUT @Path2;Jdist*12,Fomega(I)
10130 ELSE
10140 IF Cnt>15 THEN
10150 PRINT USING "14A,DDDD.DD";"FEMUR OMEGA = ";Fomega(I);" RAD/SEC"
10160 END IF
10170 Cnt=Cnt+1
10180 END IF
10190 IF Cnt>15 THEN
10200 PRINT USING "13A,DDDD.DD";"FEMUR HP = ";Fhp(I);" HP"
10210 END IF
10220 Cnt=Cnt+1
10230 END IF
10240 IF Oflag(3)=1 THEN
10250 OUTPUT @Path3;Jdist*12,Fhp(I)
10260 ELSE
10270 IF Cnt>15 THEN
10280 PRINT USING "12A,DDDD.DD";"FEMUR HP = ";Fhp(I);" HP"
10290 END IF
10300 Cnt=Cnt+1
10310 END IF
10320 CLEAR SCREEN
10330 END IF
10340 PRINT USING "11A,DDDD.DD";"FEMUR HP = ";Fhp(I);" HP"
10350 Cnt=Cnt+1
10360 END IF
0370 IF Oflag(4)=1 THEN
0380 OUTPUT @Path4;Jdist*12,Torque(I)
0390 ELSE
0400 IF Cnt>15 THEN
0410 DISP "HIT ANY KEY TO CONTINUE";
0420 ON KBD GOTO 0440
0430 GOTO 0430
0440 Cnt=0
0450 CLEAR SCREEN
0460 END IF
0470 PRINT USING "15A,DDDD.DDDD";"TIBIA TORQUE = ";Torque(I);" FT-LB"
0480 Cnt=Cnt+1
0490 END IF
0500 IF Oflag(5)=1 THEN
0510 OUTPUT @Path5;Jdist*12,Tomega(I)
0520 ELSE
0530 IF Cnt>15 THEN
0540 DISP "HIT ANY KEY TO CONTINUE";
0550 ON KBD GOTO 0570
0560 GOTO 0560
0570 Cnt=0
0580 CLEAR SCREEN
0590 END IF
0600 PRINT USING "14A,DDDD.DDDD";"TIBIA OMEGA = ";Tomega(I);" RAD/SEC"
0610 Cnt=Cnt+1
0620 END IF
0630 IF Oflag(6)=1 THEN
0640 OUTPUT @Path6;Jdist*12,Php(I)
0650 ELSE
0660 IF Cnt>15 THEN
0670 DISP "HIT ANY KEY TO CONTINUE";
0680 ON KBD GOTO 0690
0690 GOTO 0680
0700 Cnt=0
0710 CLEAR SCREEN
0720 END IF
0730 PRINT USING "11A,DDDD.DDDD";"TIBIA HP = ";Php(I);" HP"
0740 Cnt=Cnt+1
0750 END IF
0760 IF Temp=1 THEN
0770 DISP "HIT ANY KEY TO CONTINUE";
0780 ON KBD GOTO 0790
0790 GOTO 0780
0800 Cnt=Cnt+1
0810 CLEAR SCREEN
0820 END IF
0830 END SUBROUTINE
0840 END
0850 SUB Rot(Angle,Matrix(*))
0860 OPTION BASE 1
0870 Matrix(1,1)=COS(Angle)
Natrix(2,2)=COS(Angle)
Matrix(1,2)=-1*SIN(Angle)
Matrix(2,1)=SIN(Angle)
FOR I=1 TO 4
   FOR J=1 TO 4
      IF ABS(Matrix(I,J))<.000001 THEN Matrix(I,J)=0.
   NEXT J
NEXT I
END SUBROUTINE

SUB Trans(Dist(*),Matrix(*))
   THIS FORMS A TRANSLATION MATRIX THAT OFFSETS A ROTATION MATRIX
   Matrix(1,4)=Dist(1)
   Matrix(2,4)=Dist(2)
   Matrix(3,4)=Dist(3)
END SUBROUTINE

SUB Findfoot(Beta,Iota,Femur(*),Foot(*),Newfoot(*),Newfem(*))
   DIM Transl(4,4),Trans2(4,4),Temp1(4)
   MAT Transl= IDN
   MAT Trans2= IDN
   CALL Rot(Beta,Transl(*))
   MAT Newfem= Transl*Femur
   MAT Templ= Transl*Foot
   CALL Rot(Iota,Trans2(*))
   CALL Trans(Newfem(*),Trans2(*))
   MAT Newfoot= Trans2*Temp1
END SUBROUTINE

SUB Sss(S1,S2,S3,A3)
   THIS SUBROUTINE DETERMINES ANGLE 3 FOR A TRIANGLE WHERE YOU KNOW THE 3 SIDES
   P=(S1+S2+S3)/2
   Temp=SQR(P*(P-S2)/(S1*S3))
   IF ABS(Temp)>1.0 THEN
      PRINT "IMPOSSIBLE POSITION TO REACH!!"
   ELSE
A3=2*A3(1)

END IF

END SUBROUTINE

SUBEND

SUB Eqline(J(*),K(*),L(*),Marm)

THIS SUBROUTINE DETERMINES THE MOMENT ARM

OPTION BASE 1

DIM R(4)

M=(J(2)-K(2))/(J(1)-K(1))

B1=J(2)-M*J(1)

B2=L(2)+L(1)/M

R(1)=(B2-B1)/(M+1/M)

R(2)=M*R(1)+B1

Marm=SQR((R(1)-L(1))^2+(R(2)-L(2))^2)

END SUBROUTINE

END SUBEND
THIS PROGRAM WILL DETERMINE THE TORQUE NEEDED TO LEAN SKITTER THROUGH A DESIRED NUMBER OF DEGREES

THE NECESSARY TORQUE AND ANGULAR VELOCITY AT THE HIP

PROGRAM WRITTEN BY:
BRICE MACLAREN
GARY McMURRAY

OPTION BASE 1
RAD
DIM A(9,10), B(9), X(9)

THIS DATA DEFINES THE SYSTEM MATRIX THAT MUST BE SOLVED IN THE DYNAMICS TO CALCULATE THE INPUT TORQUE.

DATA 0,0,0,-1,0,1,0,0,0,0
DATA 0,0,0,0,-1,0,1,0,0,0
DATA 0,0,0,1,-1,0,0,0,0,0
DATA 0,0,1,-1,0,0,0,0,0,0
DATA 0,0,1,0,0,0,0,0,0,0
DATA 0,0,1,0,0,0,0,1,0,0
DATA 0,0,1,0,0,0,0,0,1,0
DATA 0,0,1,0,0,0,0,0,0,1
DATA 0,0,1,0,0,0,0,0,0,0

N=9

USER PROMPTED FOR VARIABLES

DISP "INPUT ANGLE OF ROTATION IN DEGREES";
INPUT Rotangle
Rotangle=Rotangle*PI/180

DISP "INPUT ACCELERATION ANGLE IN DEGREES";
INPUT Aangle
Aangle=Aangle*PI/180

DEFINE LINK LENGTHES OF 4-BAR

R=SQR(((9.1602+15.194)/12)^2+(20/12)^2)
R1=44.35
R2=R*12
R3=20
R4=20
Tlen=20/12
Flen=20/12
L=Flen
G=32.2
Mfemur=3/G
Mtibia=2/G
Wgt=100
Mbody=85/G

DEFINE INERTIA'S

CALL Invar(Mfemur, Ifemur, Flen, Mtibia, Itibia, Tlen, R, Mbody, Ibody)
CALCULATE NECESSARY ANGULAR VELOCITIES AND ACCELERATION

\[
\Phi = \text{ASN}\left(\frac{20}{(R^{12})}\right)
\]
\[
H = R \cdot \text{SIN} \left( \text{Rotangle} + \Phi + \text{Aangle} \right) - R \cdot \text{SIN} \left( \Phi + \text{Aangle} \right)
\]
\[
\text{Finalomegaad} = -1 \cdot \text{SQR} \left( \frac{2 \cdot \text{Wgt} \cdot \text{Ibody}}{L} \right)
\]
\[
\text{Alphaad} = -1 \cdot \text{Finalomegaad}^2 / (2 \cdot \text{Aangle})
\]
\[
\text{Totaltime} = \text{ABS} \left( \frac{\text{Finalomegaad}}{\text{Alphaad}} \right)
\]

DEFINE INITIAL ANGLES

\[
\text{Theta4} = \text{PI} - \text{ASN}\left(\frac{20}{(R^{12})}\right)
\]

PRINT "INITIAL THETA4 =":Theta4*180/PI

CALL Findthetas(R1,R2,R3,R4,Theta2,Theta3,Theta4)

Inittheta4=Theta4

Inittheta3=Theta3

Inittheta2=Theta2

Finalangle=Theta4-Rotangle-Aangle

BEGIN LOOPING

\[
\Delta \text{theta} = \text{Aangle} / 10
\]

FOR Temp=(Inittheta4-Deltatheta) TO (Inittheta4-Deltatheta) STEP -1*Deltatheta

\[
\text{Theta4} = \text{Temp}
\]

UPDATE ANGLES

CALL Findthetas(R1,R2,R3,R4,Theta2,Theta3,Theta4)

CALCULATE TIME

\[
T = \text{SQR} \left( \text{ABS} \left( \left( \text{Temp} - \text{Inittheta4} \right) \right) / \text{Alphaad} \right)
\]

CALCULATE ANGULAR VELOCITIES

\[
\text{Omegaad} = \text{Alphaad} \cdot T
\]

\[
\text{Omegab} = \text{R} \cdot \text{Omegaad} \cdot \text{SIN} \left( \text{Theta3} - \text{Theta4} \right) / \left( L \cdot \text{SIN} \left( \text{Theta3} - \text{Theta2} \right) \right)
\]

\[
\text{Omegabc} = \left( \text{Omegaad} \cdot \text{R} \cdot \text{SIN} \left( \text{Theta4} \right) \right) - \left( \text{L} \cdot \text{Omegab} \cdot \text{SIN} \left( \text{Theta2} \right) \right) / \left( L \cdot \text{SIN} \left( \text{Theta3} \right) \right)
\]

CALCULATE ANGULAR ACCELERATIONS

\[
\text{Num1} = \text{R} \cdot \text{Alphaad} \cdot \text{SIN} \left( \text{Theta3} - \text{Theta4} \right) - \text{R} \cdot \left( \text{Omegad}^2 \right) \cdot \text{COS} \left( \text{Theta4} - \text{Theta3} \right) + \text{L} \cdot \left( \text{Omegab}^2 \right) \cdot \text{COS} \left( \text{Theta2} - \text{Theta3} \right) + \text{L} \cdot \left( \text{Omegabc}^2 \right)
\]

\[
\text{Alphaab} = \text{Num1} / \left( L \cdot \text{SIN} \left( \text{Theta3} - \text{Theta2} \right) \right)
\]

\[
\text{Num2} = \text{Alphaad} \cdot \text{R} \cdot \text{SIN} \left( \text{Theta4} \right) + \text{Omegad}^2 \cdot \text{R} \cdot \text{COS} \left( \text{Theta4} \right) - \text{Omegab}^2 \cdot \text{L} \cdot \text{COS} \left( \text{Theta2} \right)
\]

\[
\text{Alphaab} = \text{Num2} / \left( L \cdot \text{SIN} \left( \text{Theta3} \right) \right)
\]

CALCULATE CENTER OF MASSES ACCELERATIONS

\[
\text{Accelbx} = -\text{L} \cdot \text{Alphaab} \cdot \text{L} \cdot \text{SIN} \left( \text{Theta3} \right) - \text{Omegab}^2 \cdot \text{L} \cdot \text{COS} \left( \text{Theta3} \right)
\]

\[
\text{Accelby} = \text{Alphaab} \cdot \text{L} \cdot \text{COS} \left( \text{Theta3} \right) - \text{Omegab}^2 \cdot \text{L} \cdot \text{SIN} \left( \text{Theta3} \right)
\]

\[
\text{Accelax} = -\text{L} \cdot \text{Alphaad} \cdot \text{R} \cdot \text{COS} \left( \text{Theta4} \right) - \text{Omegad}^2 \cdot \text{L} \cdot \text{COS} \left( \text{Theta4} \right)
\]

\[
\text{Accelay} = \text{Alphaad} \cdot \text{R} \cdot \text{COS} \left( \text{Theta4} \right) - \text{Omegad}^2 \cdot \text{L} \cdot \text{SIN} \left( \text{Theta4} \right)
\]

\[
\text{Accelbx} = \text{Accelbx} / 2
\]

\[
\text{Accelby} = \text{Accelby} / 2
\]

\[
\text{Accelax} = \text{Accelax} / 2
\]

\[
\text{Accelay} = \text{Accelay} / 2
\]
SUB Gauss(N,A(*),B(*),X(*))

OPTION BASE 1

FIRST, REPLACE THE LAST COLUMN OF THE A MATRIX WITH THE B MATRIX

FOR I=1 TO N
    A(I,10)=B(I)
NEXT I

FOR K=1 TO N-1
    Jj=K
    Big=ABS(A(K,K))
    Temp2=K+1
    FOR I=Temp2 TO N
        Ab=ABS(A(I,K))
        IF Big-Ab<0 THEN
            Big=Ab
            Jj=I
        END IF
    NEXT I
    NEXT K

CALL Gauss(N,A(*),B(*),X(*))

Power=X(9)*Omega

Hp=Power/550

STOP PROGRAM
FOR I=Temp2 TO N
Quot=A(I,K)/A(K,K)
FOR J=Temp2 TO N+1
A(I,J)=A(I,J)-Quot*A(K,J)
NEXT J
NEXT I
FOR I=Temp2 TO N
A(I,K)=0.
NEXT I
NEXT K
X(N)=A(N,N+1)/A(N,N)
FOR Nn=1 TO N-1
Suml=0
I=N-Nn
Ip1=I+1
FOR J=Ip1 TO N
Suml=Suml+A(I,J)*X(J)
NEXT J
X(I)=(A(I,N+1)-Suml)/A(I,I)
NEXT Nn
SUBEND

SUB Printmat(Array(*))
THIS SUBROUTINE PRINTS OUT THE INPUT MATRIX
OPTION BASE 1
FOR Row=BASE(Array,1) TO SIZE(Array,1)+BASE(Array,1)-1
FOR Column=BASE(Array,2) TO SIZE(Array,2)+BASE(Array,2)-2
PRINT USING "DDD.DD,XX,"; Array(Row,Column)
NEXT Column
PRINT
NEXT Row
SUBEND

SUB Invar(Mfemur,Ifemur,Flen,Mtibia,Itibia,Tlen,R,Mbody,Ibody)
THIS SUBROUTINE DETERMINES THE INERTIA FOR THE VARIOUS MEMBERS ABOUT THEIR AXES
Ifemur=.0252+Mfemur*(Flen/2)^2
Itibia=.01495+Mtibia*(Tlen/2)^2
IBODY IS DETERMINED ASSUMING THE BODY OF SKITTER IS SPHERICAL AND THAT THE LEGS DO NOT CONTRIBUTE SIGNIFICANTLY TO THE INERTIA
Ibody=2*Mbody*(9.16/12)^2/5+Mbody*((15.194/12)^2+(20.9817/12)^2)
RETURN TO PROGRAM
SUBEND

SUB Findthetas(R1,R2,R3,R4,Theta2,Theta3,Theta4)
THIS SUBROUTINE FINDS THE OTHER JOINT ANGLES GIVEN ONE ANGLE
FOR A 4-BAR LINK

L = SQR(R1^2 + R2^2 - 2*R1*R2*COS(PI-Theta4))

Beta = ACS((R1^2 + L^2 - R2^2)/(2*R1*L))

Psi = ACS((R3^2 + L^2 - R4^2)/(2*R3*L))

Lambda = ACS((R4^2 + L^2 - R3^2)/(2*R4*L))

IF (PI-Theta4) >= 0 AND (PI-Theta4) <= PI THEN

Theta2 = -1*(Psi-Beta)

Theta3 = PI - (PI-Lambda-Beta)

ELSE

Theta2 = -1*(Psi+Beta)

Theta3 = PI - (PI-Lambda+Beta)

END IF

RETURN TO PROGRAM