The Centipede was happy quite,
Until the Toad in fun
Said "Pray which leg goes after which?"
And worked her mind to such a pitch,
She lay distracted in the ditch
Considering how to run.

— *Mrs. Edmund Craster (American writer, d. 1874)*

5. Motion Programming

Operating the multibody passive-legged crawling vehicle, or even simulating it, requires some method of programming its motions. A motion program is a specification of motion that can be interpreted and executed so as to cause a robot to move according to the intended motion trajectories.

At the conceptual phase of this work, the motion of the crawling vehicle was described as a hierarchy, starting, at the lowest level, with an individual movement, and ascending through a leg step, a machine step, on up to a gait. An individual movement is an incremental displacement of a leg pair. A leg step is the completion of a sequence of individual movements that moves a leg pair one step forward. A machine step, or locomotion cycle, is when all of the machine's legs have each completed a step. A gait is one or more locomotion cycles that use the same distinct leg step sequence.

As discussed in Section 1.9.3, "shape control" or "spinal curve" motion programming methods might be best for performing cantilever and bridging maneuvers. However, this work focuses on the more fundamental tasks of

performing gross crawling motions over terrain. Thus, a more stride-oriented approach will be presented here.

Section 2.3 described the basic motions required to produce locomotion of the crawling vehicle. Later, Section 2.4 discussed how motion programming-related tasks could interact with the crawling vehicle's other control functions. As shown in Fig. 2.20, motion programming, for the crawling vehicle, encompasses the tasks of specifying and generating both leg pair trajectories and gaits.

Whether simulating a crawling vehicle or actually controlling a real one, the outputs of the motion program generation stage must be motion trajectories in the form of a sequence of leg pair position data sets for each leg pair. This is necessary in order to interface with the modeling techniques developed in Ch. 4, which are designed to rapidly process any arbitrary sequence of leg pair position data sets and transform them into joint space data for analysis and control.

Hence, motion trajectories are a continuous stream of 6-D position data (x, y, z, roll, pitch, yaw), which are somehow derived from the specifications of the gaits and leg pair trajectories. Specifying only a single position, or movement, at a time is cumbersome and computationally inefficient. What is needed are a higher-level programming language and a method of interpreting it. The goal of the motion programming technique developed here is to provide a simple, efficient, and versatile method of accomplishing this.

The method presented here concerns gait sequencing and the gross motions of the leg step trajectories, that is, the motions that "carry" each leg pair from one set of footholds to the next during a stride. While this method is inherently 3-D and adaptable to very rough 3-D terrain, it does not include obstacle avoidance, or terrain reactive or guarded motions intended to account for uncertainties in terrain geometry and/or kinematics. The goal of the design of this motion programming method is to build a foundation so that these more advanced motion concepts can

be added to it without having to discard the work developed here. An obstacle avoidance algorithm and a reactive footfall compensation method should be added before implementation on rough terrain.

This chapter will describe the mathematical foundation for the motion programming technique used to specify leg pair trajectories and gaits. It takes a generally bottom-up approach, beginning with a description of the leg pair trajectory specification algorithm, then presenting some new gait parameters and equations, several examples, and finally, some guidelines for programming motions.

5.1 Leg Pair Trajectories

Recall from Chapter 2 that by leg step trajectory it is meant the path of a leg pair during its step as a function of time, where its path includes both location (x, y, z) and orientation (roll, pitch, yaw).

Referring to the controls hierarchy illustrated in Fig. 2.20, desired trajectories are expressed at the trajectory specification stage using a motion programming language. This language is then interpreted at the trajectory generation stage of the hierarchy to automatically generate a time sequence of leg pair position data sets at the controller setpoint update rate for each leg pair in motion. This section elaborates on the trajectory specification and generation techniques that were introduced in Sections 2.6.1.7 and 2.6.1.8 and describes their mathematics.

5.1.1 Desirable Traits for the Leg Pair Trajectory Specification Algorithm

For good performance, the *algorithm* used to specify, store, and generate the motion trajectories used with the crawling vehicle must embody certain desirable traits. But first, recall from Section 2.6.1.7 that the motion trajectories themselves have their

own operational constraints and desirable characteristics. Briefly stated, the resulting trajectories should produce reasonable robot movements in Cartesian space (the real world where the robot is trying to operate). That is, the trajectories should be energy-efficient, collision-free, and at all times within the workspaces of the Stewart-Gough Platform mechanisms.

The algorithm for producing such motion trajectories must be carefully designed. Integral to the trajectory specification method is a programming language adapted for the expression of motion commands. As with other written languages, this language enables information to be written, stored, and read. More exactly, the motion programming language is used to specify motions, store sequences of motions, and enable these stored motions to be interpreted at a later time. To function well, the language should be:

- Easy to write — The language should provide for simple specification of motion trajectories. This reduces complexity and increases the execution speed of the trajectory specification system.

- Concise — The language should express motion program concepts succinctly, so as to reduce the amount of computer memory required to "write" it down. This compact storage will speed network transmission of the leg step trajectories. For example, once the "head" processor has determined the next set of footholds and the step trajectory for a leg pair, a compact expression of the stride trajectory can be transmitted rapidly to the leg pair processor, which will control execution of the trajectory.

- Easy to "read" — The language should allow rapid and unambiguous interpretation, so that trajectory generation at runtime will be error-free and rapid enough to keep up with the desired control update rate of the leg pair controllers. Also, a simple, human-readable format would simplify software debugging.

- Versatile — The language's method of expression should not artificially limit the usable workspace of the robot.

As explained in Chapter 2, to facilitate obstacle avoidance and the control of antagonistic (and possibly contentious) actuators, motion trajectories will be initially specified in Cartesian rather than joint space.

5.1.2 Trajectory Curve Selection

The geometry of the stride path can be specified using the following algebraic forms: explicit, implicit, parametric, and intrinsic (Mortenson, 1985). However, the explicit and implicit forms have been found to be relatively inefficient for evaluation using computers, making them poor choices for use with the robot's real-time digital control system. Similarly, evaluation of the intrinsic form traditionally involves a numerical solution of the Frenet-Serret equations (Struik, 1950). While the intrinsic form offers simple determination of path curvature and torsion, at this stage of the research, these parameters do not seem necessary. Thus, solution speed must be given priority, precluding the use of the intrinsic form. This leaves us with the parametric representation of curves, which has been the preferred form for use in computer graphics and Computer-Aided Design programs (Mortenson, 1985). The parametric form gets its name because it includes an independent parameter, usually indicated by "u" and referred to as the parametric variable.

Recalling that the trajectory is not just a path, but a sequence of positions along the path as a function of time, we see that the parametric variable is convenient for describing the leg step trajectories because the parametric variable in *itself* can be made a function of time.

Having chosen the basic algebraic form, we next must choose the types of curves used to define the stride path geometry. Hermite cubic curves are used in this work

for the motion specification for several reasons. First, the cubic form is observed in many natural shapes and phenomena. Second, the mathematical formulation of parametric cubic curves is well developed because of their use in computer graphics. Third, if a single cubic curve segment does not provide sufficient flexibility for the stride path shapes, multiple cubic segments can be combined together end to end to enable modeling more complex paths having more than two changes in direction. When creating these compound curves, the tangency and continuity of curvature at the intersections can be ensured utilizing techniques developed for computer graphics. Compound curves can also be reparametrized to simplify their use as a unit.

Therefore, in this work, Hermite parametric cubic curves have been used to specify the motion of each leg pair in both *xyz* space and roll-pitch-yaw space. Specifically, each trajectory segment is defined by two coupled 3-D parametric cubic curves. One curve specifies the *xyz* translational motion of the leg pair as a function of the parametric variable, while the other curve specifies the roll-pitch-yaw rotational motion of the leg pair as a function of the parametric variable. The two curves are said to be coupled because they share the *same* parametric variable, which is a function of time. Hence, together they specify a leg pair trajectory for a certain span of time that depends upon the times associated with the beginning and ending values of the parametric variable. Note that it would also be valid to combine the parametric cubic curves together and refer to this method as a six-dimensional (6-D) trajectory specification method. The following two sub-sections explain how parametric cubic curves have been used to define leg step trajectories in this research.

5.1.3 Specifying Leg Pair Location in XYZ Space

Hermite parametric cubic space curves are used to specify the location of each leg pair as a function of time during its stride or "leg step". The equation for defining each point along a leg step trajectory is as follows:

$$\mathbf{P}_{XYZ}(u) = \mathbf{F}\,\mathbf{B}_{XYZ} \tag{5.1}$$

Where the **F** matrix contains the so-called "blending functions" and **B** is a 4x3 matrix of geometric coefficients or boundary values (Mortenson, 1985). The blending functions are defined by:

$$\mathbf{F} = \begin{bmatrix} u^3 & u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{5.2}$$

Note that the 1x4 vector includes the cubic function of the parametric variable, *u*, and the 4x4 matrix is referred to as the "universal transformation matrix" (Mortenson, 1985).

The boundary conditions are defined as follows:

$$\mathbf{B}_{XYZ} = \begin{bmatrix} \mathbf{P}(0) \\ \mathbf{P}(1) \\ \dfrac{d\,\mathbf{P}(0)}{du} \\ \dfrac{d\,\mathbf{P}(1)}{du} \end{bmatrix} \tag{5.3}$$

In Eq. 5.3, **P**(0) is the cubic curve evaluated at $u = 0$ (i.e. the starting point $\begin{bmatrix} p_x & p_y & p_z \end{bmatrix}$ of the trajectory curve). Similarly, **P**(1) is the curve evaluated at $u = 1$ (the end point of the curve). Vectors $d\mathbf{P}(0)/du$ and $d\mathbf{P}(1)/du$ are the tangent vectors at the beginning

and end of the curve, respectively (see Fig. 5.1). The directions and magnitudes of the two tangent vectors are chosen so that the trajectory will avoid any obstacles between the starting and ending points of the curve.

In the current simulation, these coefficients are expressed relative to the local coordinate frame, the same frame that would be used for the Local Terrain Map. In the future, it may be desirable to add the ability to also express the coefficients relative to a particular leg pair's present position. (As explained in Appendix C, this latter method may help in certain teleoperation functions.)

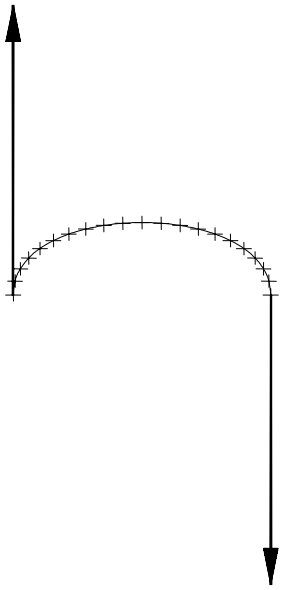

Figure 5.1 – Parametric Cubic Curve with Tangent Vectors

5.1.4 Specifying Leg Pair Orientation in Roll-Pitch-Yaw Space

Hermite parametric cubic space curves are also used to specify the orientation of each leg pair as a function of time during its leg step. The equation for orientation is

essentially the same as that for location (Eq. 5.1), except that the boundary conditions are changed.

$$\mathbf{P}_{\gamma\beta\alpha}(u) = \mathbf{F}\ \mathbf{B}_{\gamma\beta\alpha} \tag{5.4}$$

This equation has the exact same blending functions, **F**, as the *xyz*-space equation presented in the previous section (Eq. 5.2). Similarly, the geometric coefficient matrix has the same basic form as before. The only difference is that rather than specifying the leg pair frame's *xyz* locations as a function of *u*, the new geometric coefficients specify the roll, γ, about the **X**-axis, the pitch, β, about the **Y**-axis, and yaw, α, about the **Z**-axis as functions of *u*. As with the *xyz* coefficients of Eq. 5.3, these angles are measured relative to the local coordinate frame, *L*.

$$\mathbf{B}_{\gamma\beta\alpha} = \begin{bmatrix} \mathbf{\Phi}(0) \\ \mathbf{\Phi}(1) \\ \dfrac{d\,\mathbf{\Phi}(0)}{du} \\ \dfrac{d\,\mathbf{\Phi}(1)}{du} \end{bmatrix} \tag{5.5}$$

$\mathbf{\Phi}(0)$ is a 1x3 vector that specifies the roll, pitch, and yaw orientations of the frame at the beginning of the trajectory curve ($u = 0$). Similarly, $\mathbf{\Phi}(1)$ specifies the orientations at the end of the curve. Tangent vectors $d\,\mathbf{\Phi}(0)/du$ and $d\,\mathbf{\Phi}(1)/du$ specify the rates of rolling, pitching, and yawing at the beginning and end of the leg step, respectively.

5.1.5 Simple Leg Steps

As described in the previous two subsections, a combination of two 3-D parametric cubic curves defines all 6 DOF's of a leg pair *trajectory segment*. A simple leg step can be specified using a *single* trajectory segment. Using this method, specifying a leg step amounts to assigning the boundary conditions of the two parametric cubic curves (Eq. 5.3 and 5.5).

For a simple leg step, the starting point, $\mathbf{P}(0)$, of the curve (see Eq. 5.3) corresponds to the leg pair's foothold location before lifting off of the ground, and the ending point, $\mathbf{P}(1)$, corresponds to the leg pair's foothold location after returning to the ground at the end of the stride. The tangent vectors $d\mathbf{P}(0)/du$ and $d\mathbf{P}(1)/du$ are effectively the angle of take-off for the leg step as the leg pair initially lifts off of the ground and the angle of approach as the leg pair returns to the ground, respectively. Note that the tangent vectors can be in *any* arbitrary direction in 3-D, enabling this technique to describe locomotion over rough terrain. Specifically, this will allow the take-off and landing trajectories of the leg steps to be normal to the ground surface, regardless of the slope of the ground.

Similarly, looking at Eq. 5.5, the $\mathbf{\Phi}(0)$ vector specifies the initial orientation of the leg pair frame at the start of the stride, and $\mathbf{\Phi}(1)$ specifies the orientation after the leg pair has returned to the ground at the end of the step. Tangent vectors $d\mathbf{\Phi}(0)/du$ and $d\mathbf{\Phi}(1)/du$ specify the rotational rate of change with respect to the parametric variable at the beginning and end of the leg step, respectively. As with the curve for *xyz*-position, the 3-D nature of these orientation curves will enable the leg pairs to be programmed to rotate so that the beginning and end of the leg step can be made to conform to a wide variety of sloping 3-D terrain.

Recall that the boundary conditions of the parametric cubic curves are specified relative to the local coordinate frame, *L*. For the *xyz* curve, this is straightforward.

However, because rotations are not generally commutative (Craig, 1988), specifying the coefficients for the $\mathbf{\Phi}$ vectors can require additional calculations. For example, if it is desired to rotate a leg pair about its own ankles, but the leg pair frame is not oriented the same way as the local frame, then simply specifying a pitch angle relative to either the local frame or the leg pair frame will *not* yield the desired result. In this case, which may be common (as discussed in the next section), the rotation must first be defined relative to the leg pair's current frame to create a new transform, ${}^{l}_{l'}\mathbf{T}$, from the desired position of the leg pair, *l'*, to the current position of the leg pair, *l*. (The equation for this, Eq. C.1, is provided in Appendix C). Next, this new transform for the leg pair must be converted to be with respect to the local coordinate frame, as follows:

$$ {}^{L}_{l'}\mathbf{T} = {}^{L}_{l}\mathbf{T}\ {}^{l}_{l'}\mathbf{T} \tag{5.6} $$

Finally, to determine the values for $\mathbf{\Phi}(1)$, we must extract the equivalent local frame roll, pitch and yaw angles from the transformation matrix resulting from Eq. 5.6. Assuming that the transform matrix elements are numbered starting with (1,1) at the top left corner, the equations for extracting the equivalent local frame roll, pitch and yaw angles from the leg pair frame are (Craig, 1988):

$$ \begin{aligned} \gamma &= \mathrm{Atan2}\left(\mathbf{T}_{32}, \mathbf{T}_{33}\right) \\ \beta &= \mathrm{Atan2}\left(-\mathbf{T}_{31}, \sqrt{\mathbf{T}_{11}{}^{2} + \mathbf{T}_{21}{}^{2}}\right) \\ \alpha &= \mathrm{Atan2}\left(\mathbf{T}_{21}, \mathbf{T}_{11}\right) \end{aligned} \tag{5.7} $$

In Eq. 5.7, the subscripts denote the matrix elements. Also note that the $\mathrm{Atan2}(y,x)$ function is the two-argument arc tangent function, which yields correct answers over

the full 360° range. These computed values for the roll, pitch, and yaw can then be used in the $\mathbf{\Phi}(1)$ vector to yield the desired rotational motion.

Hence, specifying a simple leg step composed of a single trajectory segment entails assigning the 8 vectors (for a total of 24 scalar values) shown in Eq. 5.3 and 5.5.

Choosing all these values is simpler than it may initially appear, because, as discussed earlier in this section, $\mathbf{P}(0)$ and $\mathbf{\Phi}(0)$ are already specified for us, since they represent the current position of the leg pair, thus reducing the number of values to be determined by 6. Similarly, the values of $\mathbf{P}(1)$ and $\mathbf{\Phi}(1)$ specify the position of the leg pair at the end of the leg step and are derived from the chosen footholds. (Recall that foothold selection was discussed in Section 2.6.1.6.) Specifically, the location of the leg pair coordinate frame origin defines $\mathbf{P}(1)$, and the slope of the terrain underfoot, along with the direction of the local path at that location, define $\mathbf{\Phi}(1)$. Furthermore, for simple leg steps, the angular velocities of the leg pair at the beginning ($d\mathbf{\Phi}(0)/du$) and end ($d\mathbf{\Phi}(1)/du$) of the step will always be zero, hence reducing the number of undefined boundary values by another 6. Last, the take-off and landing tangents, ($d\mathbf{P}(0)/du$) and ($d\mathbf{P}(1)/du$), are specified to be normal to the terrain underfoot at the beginning and ending of the leg step, respectively. Thus, their unit vectors are defined, and we are left with the choice of their magnitudes. These choices are trade-offs between obstacle avoidance safety and energy efficiency. Specifically, the tangent magnitudes should be as small as allowable to prevent unnecessary lifting but large enough to produce a trajectory that lifts the feet high enough to avoid colliding with any terrain features that may exist between the starting and ending footholds.

Thus, once the footholds along the local path are selected and the underlying ground slope is estimated, there are actually only 2 remaining scalar values to be chosen from the original 24, the magnitudes of the take-off and landing tangent vectors. If choosing these to accomplish obstacle avoidance becomes problematic, it

may be helpful to use the four-point form of parametric cubic space curves instead of the Hermite form discussed here (Mortenson, 1985).

Once a leg pair trajectory has been specified, it can be *generated* at runtime by evaluating Eq. 5.1 and 5.4 for a series of time steps. Specifically, at each time increment, the associated u value is substituted into Eq. 5.1 and 5.4 to produce a [x y z γ β α] data set, which defines the position of the leg pair at the corresponding time. (While this computation might seem complicated at first, it must be remembered that it has been used successfully in CAD programs for a number of years.) The resulting data sets can then be used for inverse kinematics, stability determination, and rendering via the modeling techniques presented in Ch. 4.

5.1.6 Compound Leg Steps

In the investigation of caterpillars presented in Ch. 3, it was observed that caterpillars pitch their legs forward both *before* they lift them off of the ground and *after* they have placed them back down.

The simple leg step formulation presented in the previous section enables *simultaneous* pitching and curvilinear motion, but it cannot, by itself, specify pitching *followed by* curvilinear motion. Thus, as described in the previous section, while a functional leg step trajectory can be specified for a leg pair by using a single trajectory segment composed of a coupled pair of 3-D parametric cubic curves, it can only *approximate* a true caterpillar "leg step".

What is needed, in addition, is a way to specify pitching rotations about the leg pair ankles both before and after the transfer phase of the stride. Recall from Section 4.2.1 that the origin of each leg pair coordinate frame was moved to the centerpoint between the ankle joints. This enables a parametric cubic trajectory segment to be

reduced to a simple pitching motion by simultaneously specifying pure rotation about the **Y**-axis of the leg pair coordinate frame and no translation in *xyz*.

Hence, the approach taken in this work was to rely on the versatility of the coupled parametric cubic form, and to specify these pitching leg steps by simply using three trajectory segments instead of one. The first segment of such a compound leg step would specify the rotation angle about the ankles before lift-off. The second segment would simultaneously specify the backward pitching and the *xyz* motion to move the leg pair to its next footholds. And the third segment would specify the forward pitching about the ankles (the feet now being back in contact with the ground) to return the leg pair to an upright position.

This approach is advantageous in that it does not require additional software development, lengthen the robot's control program code, or introduce additional complexity into the motion programming language. A disadvantage is that there are other, more concise, ways that the pitching motions could have been expressed within the motion programs.

5.2 Gaits

As discussed in Ch. 3, the use of appropriate gaits (sequence plans for lifting and placing legs) is essential to achieve the performance potential of legged vehicles. Song and Waldron (1986) emphasized that the choice of gait has a strong effect on a walking vehicle's geometric design, mobility, control algorithm, and efficiency.

While other gaits are possible for use with the crawling vehicle, wave gaits will be used here because, as explained in Section 3.3, caterpillars appear to use them exclusively.

5.2.1 Traditional Gait Analysis Definitions

Wave gaits, by definition, are periodic. Previous researchers of legged locomotion have used a number of basic parameters to help describe periodic gaits. According to Song and Waldron (1989), most of these definitions were established by McGhee (1985) and his colleagues. A pertinent selection of these gait analysis definitions are quoted, as follows, from Song and Waldron (1989), with slight modifications for clarity.

- The number of legs for a legged locomotion system is denoted symbolically by n.
- The stride or stride length, λ, is the distance the center of gravity of the vehicle translates during one complete locomotion cycle.
- The cycle time, T, is the time for a complete cycle of leg locomotion of a periodic gait.
- The duty factor, β_i, is the time fraction of a cycle time in which leg i is in the support phase (on the ground supporting the vehicle). It is assumed to be the same for all the legs.

$$\beta_i = \frac{\text{time of support phase of leg } i}{\text{cycle time of leg } i}$$

- The leg phase, ϕ_i, is the fraction of a cycle period by which the contact of leg i on the ground lags behind the contact of leg 1 (where leg 1 is generally assumed to be the frontmost left leg).

In addition, Waldron and his colleagues (1984) describe:

- The transfer phase time, or return time, τ, of a leg as the period of time during a step in which the foot is not on the ground.

In the interest of avoiding confusion, the same variable names and symbols have generally been used in this work. However, because of the differences between the multibody passive-legged crawling vehicle and the single-body active-legged walking vehicles for which these parameters were originally defined, and because of the differences between this research and the earlier studies, it is desirable to redefine one parameter and replace two others with new parameters better suited for the work at hand.

The parameter, n, as traditionally defined, is awkward for use with the crawling vehicle. This is because the traditional gait parameters were defined for animals and robots that used out-of-phase symmetry (lateral legs moving 180° out of phase with respect to each other), whereas, with the crawling vehicle, we are exploring in-phase symmetric gaits similar to those exhibited by caterpillars. Thus, it is desirable to redefine the n parameter slightly to make it better suited for describing this case. Specifically, if we interpret n to be the number of independent support units, then, in the case of this robot, which uses in-phase symmetric motion programs, instead of individual legs,

- n, stands for the number of *leg pairs*.

However, simply substituting leg pair for leg into the old equations is still not ideal. The customary practice of defining a periodic gait by specifying a duty factor, β, and phase, ϕ, for each leg (or leg pair) is awkward in the case of this research. This is because the values of β and ϕ are defined relative to the overall locomotion cycle and are dependent upon the total number of legs, or leg pairs. While these parameters have proven useful for gait analysis studies, they are awkward for gait specification and generation, which are our purposes here. Specifically, using leg phase and duty factor, a gait that produces a certain relative timing of longitudinally adjacent leg pairs on a six leg pair robot would have a *different* mathematical definition than a gait used on an 8 leg pair robot that produces the *same* relative

motions of its longitudinally adjacent leg pairs. Thus, their use is awkward because, in this research, the number of leg pairs is an important design variable, and functionally equivalent gaits should not have different mathematical definitions just because a longer or shorter crawling vehicle is using them.

Hence, the goal of replacing these parameters is to make the gait-defining parameters more independent of the number of leg pairs in a particular robot. This will enable direct comparison of gait formulae between crawling vehicles with dissimilar numbers of leg pairs.

With the crawling vehicle, the choice of gaits is restricted to wave-type gaits that use in-phase symmetry of lateral legs and that start moving at the rearmost leg pair (so as to maximize longitudinal stability). As a result, we always know that the next leg pair to move will be the next-most anterior one. This relative simplicity, when compared to general rigid-body active-legged walking vehicles, allows the use of a simpler method of defining gaits that is more convenient for motion programming and is independent of the number of leg pairs. Specifically, this research proposes specifying gaits by replacing leg phase and duty factor with new parameters called the "step interval" and "wave interval".

5.2.2 The Step Interval

The "step interval", σ, is the ratio of the time delay from the beginning of the ith leg pair's leg step until the beginning of the leg step of leg pair $i-1$, to the total duration of the leg step of leg pair i. For example, if a gait calls for leg pair 4 to move through three-quarters of its leg step before starting the leg step of leg pair 3, then it has a step interval of $\sigma = 0.75$. The step interval value effectively determines the number of leg pairs and actuation units that move concurrently during an individual "wave" of motion. It is analogous to a pulse width in signal processing.

The step interval is defined within the range $0 < \sigma \leq \infty$ and is non-dimensional. The peculiar case of $\sigma = 0$ corresponds to a robot hopping simultaneously with all of its leg pairs. However, this case is excluded here because it is not a wave gait (its leg steps being simultaneous rather than sequential). Gaits with $\sigma > 1$ all have, from the standpoint of gait definition, the same sequence of steps, with only a variation in the separation between the discrete steps. At this stage of the research, it appears that step interval values in the range of $0.5 \leq \sigma \leq 1.1$ are the most likely to have practical usefulness with the multibody passive-legged crawling vehicle.

5.2.3 The Wave Interval

As mentioned in Ch. 3, the caterpillars were sometimes observed to travel by means of more than one wave of motion on their body at the same time. When moving rapidly, the caterpillars appeared to have approximately 2.5 simultaneous waves on their 12 segment bodies. Therefore, it is desirable to have a standard method for specifying this multiple wave locomotion for the crawling robot.

The method used in this research is to define a "wave interval", ω, which is the separation between consecutive waves of leg steps. The wave interval is enumerated in terms of the number of step intervals of the current wave that are performed before starting the next wave of leg steps. For example, an $\omega = 4$ would mean that a new wave would start at the posterior leg pair after the current wave has completed the step intervals of four leg pairs. The wave interval, ω, can be defined between $\frac{1}{\sigma}$ and ∞, where the $\frac{1}{\sigma}$ constraint precludes the impossible situation of defining a leg pair as being simultaneously part of two successive waves.

When $\omega = n$, then the vehicle will have exactly one wave of leg steps active at a time — just as one wave reaches the front of the robot, the next begins at the rear. When $\omega < n$, then the vehicle will have more than one wave of locomotion active

simultaneously (thus imitating the multiwave locomotion of caterpillars). Values of $\omega > n$ can cause pauses between each wave of locomotion, making each locomotion cycle "discrete". By a discrete gait, it is meant that there is no overlap in the execution of consecutive cycles of locomotion; each wave completely finishes and there is a pause before the next wave starts. If the pitching motions of leg pairs that are on the ground are ignored, then a locomotion cycle is discrete whenever its wave interval is greater than or equal to:

$$\omega_d = n + \frac{(1-\sigma)}{\sigma} \tag{5.8}$$

Where the second term in Eq. 5.8 accounts for the remainder of the frontmost leg pair's stride after it has completed the step interval portion of the stride. In the general case, including any pitching motions of leg pairs that are on the ground, a locomotion cycle is discrete whenever its wave interval is greater than or equal to:

$$\omega_d = n + \frac{(1-\sigma)}{\sigma} + \frac{t_{pb} + t_{pe}}{\tau\sigma} \tag{5.9}$$

Where the third term of the equation accounts for compound leg steps. In this equation, t_{pb} is the duration of the pitching motion at the beginning of the leg steps, and t_{pe} is the duration of the pitching motion at the end of the leg steps. Since the transfer phase time, τ, has the same units as t_{pb} and t_{pe}, their units cancel, and the equation is unitless.

Thus, when gaits have $\omega \geq \omega_d$, each of their locomotion cycles is discrete and there are time delays between the successive cycles or waves. Equation 5.8 is used when dealing with only simple steps or when we are concerned with only the gait. (Recall that the underlying gait of a motion program is concerned with only the sequence of foot lifting and foot placing events, not with pitching motions of leg pairs on the

ground.) Equation 5.9 is used when there are compound leg steps and it is desired to account for *all* the motion pertaining to the waves.

At this stage of the research, it is expected that the most useful values for the wave interval will be in the range: $\frac{n}{2} \le \omega \le n$. However, the ranges of allowable values discussed here for σ and ω do not ensure that a particular gait will be useful or even stable for a particular robot design. The evaluation of the performance of various gait and vehicle combinations is the purpose of the simulation tools to be described in the next chapter.

Using the new notation presented here, the caterpillars described in Ch. 3 can be said to have exhibited locomotion in the range: $\sigma \approx 0.5$ and $4.8 \le \omega \le 12$. Therefore we can dispense with confining terms, such as "single leg pair per wave gait" and "double leg pair per wave gait" that were used in Ch. 3, and simply say, $\sigma = 1.0$ and $\sigma = 0.5$. The new notation can clearly describe the continuum of possible leg step sequences, making it much less awkward to study arbitrary wave gaits, such as a $\sigma = 0.634$, $\omega = 4.278$ gait.

5.2.4 Timing Considerations

Note that in the above discussions of the step interval and the wave interval that both parameters are unitless. This means that from the gait theory point of view, these gait definitions are independent of time and can be described and analyzed without reference to time. This underlies the fact that it is possible to use the same gaits at different locomotion speeds. However, once a time value has been assigned to the transfer phase time variable, τ, we can readily convert the gait definitions into time values that describe the sequence of lift-offs and placements of the feet of all the leg pairs. (Thus, in the parlance of Meisel and McGhee (McGhee, 1968) accomplishing "gait realization".) These computed time values will inherit whatever time units were used to define τ.

Specifically, once the transfer phase time, τ, is known, the step interval delay (that is, the time delay between successive leg steps) can be computed from the relation:

$$\delta t_s = \tau\sigma \tag{5.10}$$

Again, using the transfer phase time, τ, the time delay between successive waves can be computed from the equation:

$$\delta t_w = \tau\sigma\omega \tag{5.11}$$

Thus, another way of thinking of the wave interval is that each unit of ω corresponds to an amount of time equal to $\tau\sigma$.

For discrete locomotion cycles ($\omega > \omega_d$), the pause between the cessation of motion of one wave and the beginning of motion of the next wave (including the pitching motions of compound leg steps) is:

$$t_d = (\omega - \omega_d)\tau\sigma \tag{5.12}$$

Finally, when the transfer phase time, τ, is held constant for an entire locomotion cycle, the resulting cycle time is:

$$T = \tau\sigma n \tag{5.13}$$

5.2.5 Vehicle Speed

The speed of wheeled vehicles is simply determined by wheel diameter and rotation speed (assuming no slip). The speed of the multibody passive-legged crawling vehicle is determined by several different factors. Understanding the components or parameters that determine vehicle velocity is important, since locomotion speed is an important measure of performance for comparing crawler designs and motion programs.

Each locomotion cycle for the crawling vehicle can be thought of as a wave of leg pair motion that ripples from the rear of the robot to the front. This rear to front direction was observed in caterpillars and maximizes longitudinal stability. The speed of the vehicle can be controlled by modulating three parameters, namely, the wave speed, the wave size, and the time interval between waves. Using the new notation, these controlling parameters are specified by τ, λ, σ, and ω. The stride length, λ, together with the transfer phase time, τ, determine the wave speed by specifying the distance moved and the duration of the transfer phase for each stride. The step interval, σ, governs the number of leg pairs and actuation units that simultaneously move as part of each individual wave, and hence the wave size. And finally, the wave interval, ω, specifies a period in terms of $\tau\sigma$ that determines the time delay between successive waves.

For the case of a single wave ($\omega = n$) gait, the equation for the average speed of the robot center of gravity is:

$$s_{cg} = \frac{\lambda}{\tau\sigma n} \tag{5.14}$$

Extending Eq. 5.14 by including the wave interval, the general-case speed equation for a wave gait is as follows:

$$s_{cg} = \left(\frac{\lambda}{\tau\sigma n} \right) \frac{n}{\omega} = \frac{\lambda}{\tau\sigma\omega} \tag{5.15}$$

Equation 5.15 holds so long as $\sigma > 0$ and $\omega \geq 1/\sigma$. It is valid, both when wave gaits are used continuously, and when they are broken up into discrete locomotion cycles. Note that the speed, s_{cg}, is the velocity of the robot center of gravity averaged over a fully developed locomotion cycle. The idea of full development of a gait is best illustrated with a graph. Figure 5.2 plots the center of gravity location versus time for a robot starting to crawl from a stationary position.

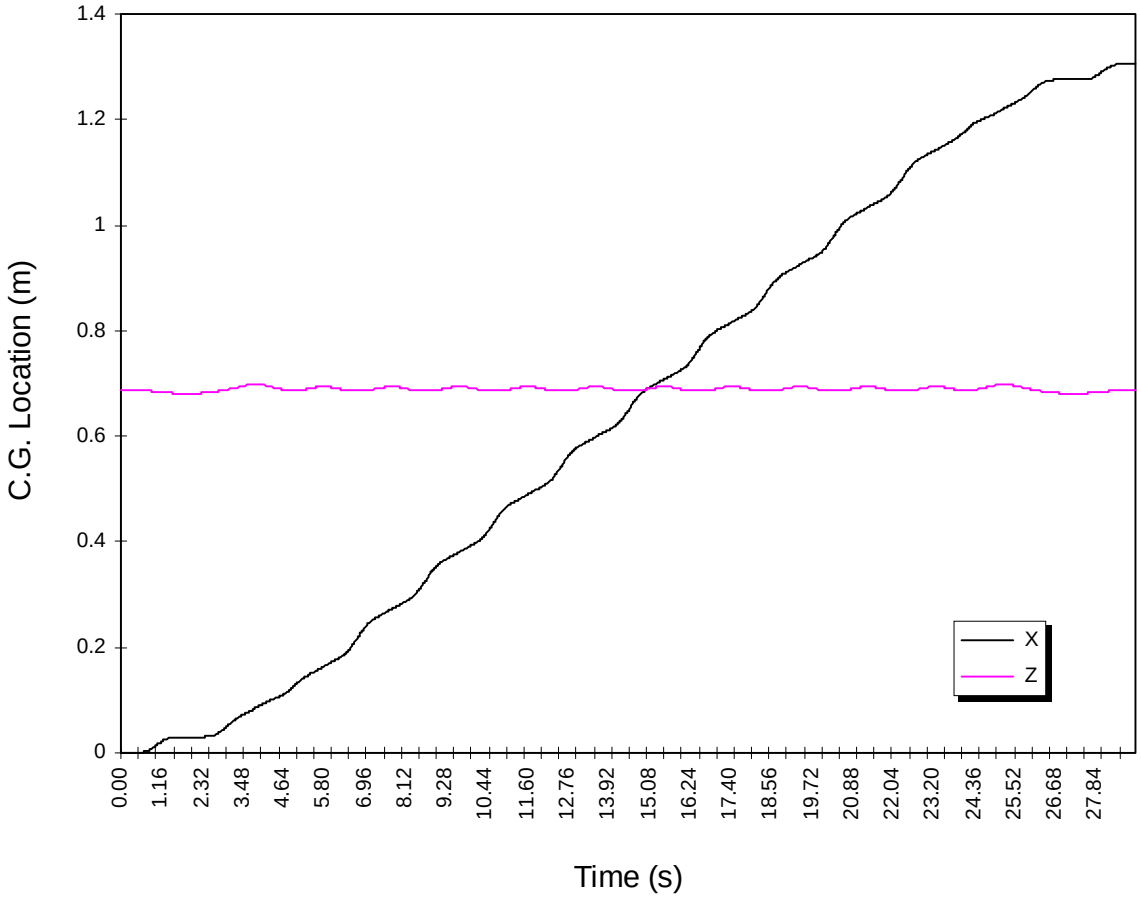

Figure 5.2 – C.G. Location vs. Time for a 6 Leg Pair Robot Performing a $\sigma = 0.52, \omega = 6.0$ Gait for Two Locomotion Cycles, Starting from and Ending with a Complete Stop.

The simulation example used to create Fig. 5.2 is of a vehicle with 6 leg pairs that moved forward over flat, smooth terrain. The vehicle was programmed such that, starting from a motionless state, it crawled straight along the **X**-axis using gait parameters of $\sigma = 0.52$ and $\omega = 6.0$ for exactly 2 locomotion cycles, and then it completely stopped.

The generally horizontal curve in Fig. 5.2 is the *z*-value, or elevation, of the robot's center of gravity. It is basically horizontal because the robot is moving over flat,

smooth terrain. As each leg pair performs its leg step, the C.G. oscillates slightly up and down.

More significant to our discussion is the generally diagonal curve in Fig. 5.2, which shows the forward motion of the C.G. along the **X**-axis. Because the vehicle is starting from a complete stop, and since after the 6th leg pair starts its first leg step there is a delay of $\tau\sigma$ before the 5th leg pair begins to move, it takes some time for the vehicle to reach full speed. For similar reasons, at the end of the second locomotion cycle, the vehicle slows before coming to a full stop. These accelerations and decelerations are a function of the gait, not of any actuator torque limitations. Looking at the graph, it can be seen that the vehicle does not reach its "fully developed" gait speed until about 7 seconds into the motion program and then drops off this speed at about 22 seconds. Thus, in this example, the speed value, s_{cg}, calculated in Eq. 5.15 would correspond to the average slope of the *x* curve on the interval between approximately $t = 7\,\text{s}$ and $t = 22\ \text{s}$.

5.2.6 Example Gait Diagrams

In order to illustrate the effects of varying the step interval and wave interval, gait diagrams of five different motion program examples are presented in Fig. 5.3 through 5.7. In the figures, time is marked-off in units of $\tau\sigma$, where the transfer phase time, τ, can be any arbitrary finite amount of time and can be expressed using any time units. Once τ is selected, the timing of all the events of the motion programs can be determined. But the exact timings make no difference from the gait theory viewpoint, because the gait definitions are independent of stride speed.

5.2.6.1 A Simple Example Motion Program (σ = 1.0, ω = n)

The example gait diagram in Fig. 5.3 depicts a case of $\omega = 4 = n$. As each wave of leg steps finishes at leg pair 1, it appears to "wrap around" from the front of the vehicle to the rear to start the next wave at leg pair 4.

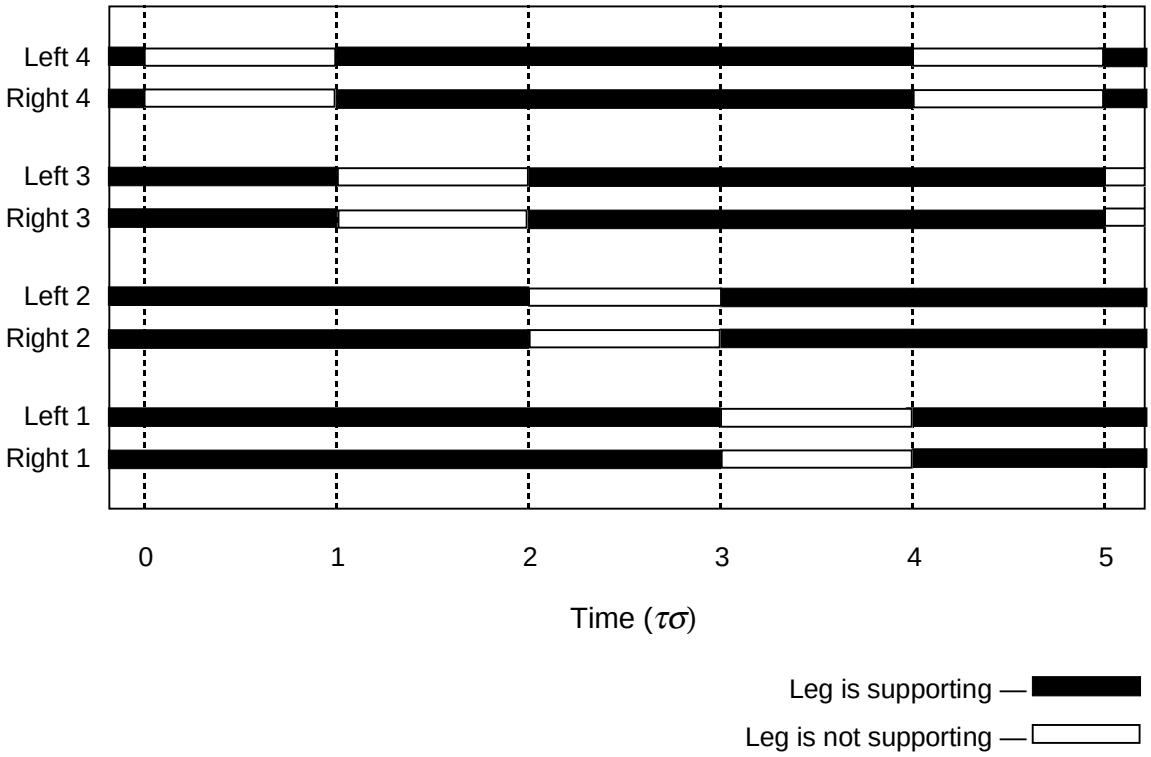

Figure 5.3 – Gait Diagram of a $\sigma = 1.0, \omega = 4.0$ Wave Gait, Performed on a 4 Leg Pair Robot.

The $\sigma = 1.0$ gait has exactly one leg pair in motion at any time. This gait is singular because one leg pair returns to the ground at exactly the same time as another leg pair takes off.

5.2.6.2 An Example with Overlapping Leg Steps (σ = 0.5, ω = n)

Figure 5.4 shows a gait diagram of a robot with 6 leg pairs, in contrast to the 4 leg paired robot of the previous example. Nevertheless, from the gait point of view, the main difference is that the motion program in this new example uses a step interval that is less than one. As a result, its leg steps overlap in time. This $\sigma = 0.5$ gait is similar to the gait observed in use by the caterpillars. Back in Ch. 3 it was termed the "double leg pair per wave" gait, since two leg pairs are in motion at a time for each wave.

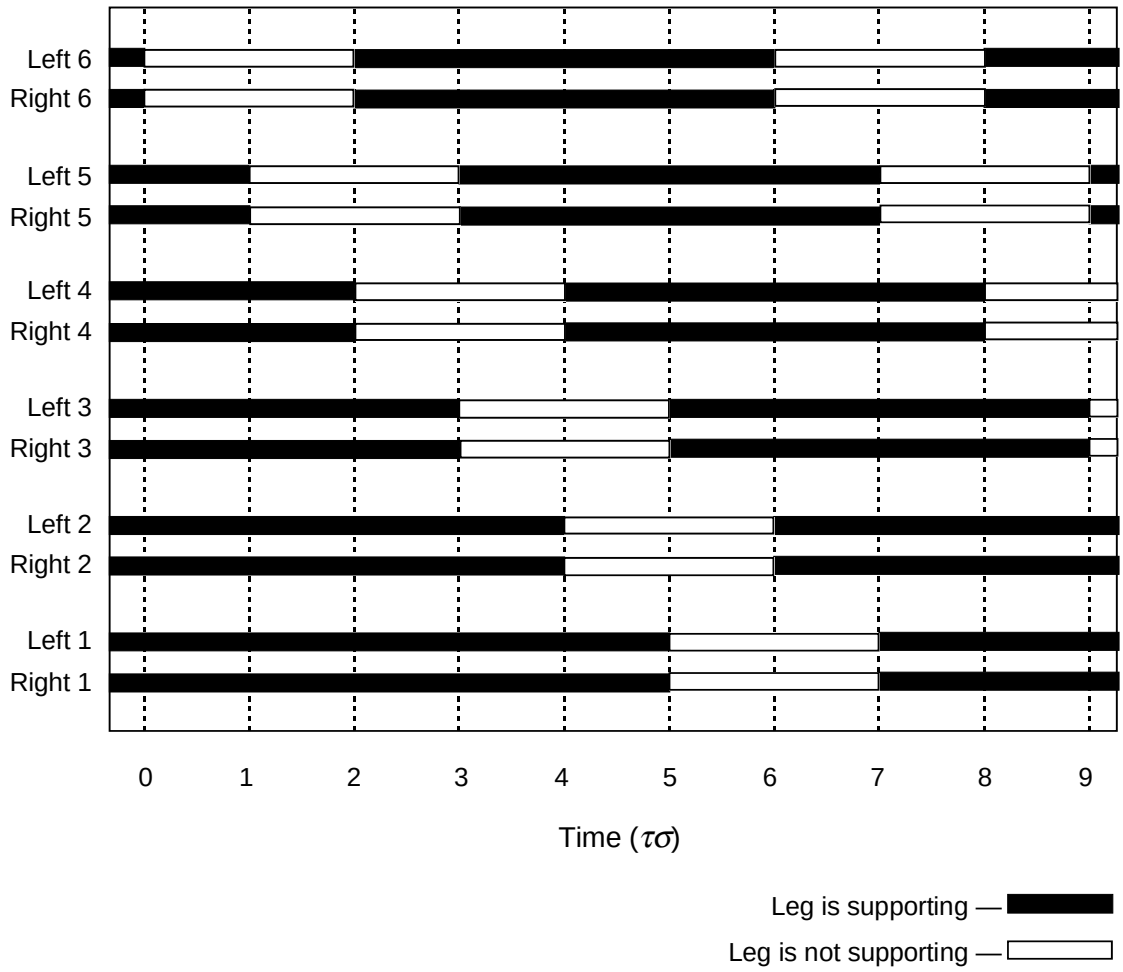

Figure 5.4 – Gait Diagram for a $\sigma = 0.5, \omega = 6.0$ Gait, Performed on a 6 Leg Pair Robot.

Once again, because $\omega = n$, the second wave of leg steps begins just as the first wave ends. Looking at Fig. 5.4 at $t = 6\tau\sigma$, while leg pair 1 has not completed its full leg step, it *has* completed its step interval before leg pair 6 starts the second wave. (Recall that the wave interval is enumerated in terms of step intervals rather than completed leg steps.) We see that the step interval from leg pair 1 to leg pair 6 is the same as the step interval between leg pairs 3 and 2, or any of the other adjacent leg pairs. Thus, *exactly* one wave is in progress at all times.

The $\sigma = 0.5$ gait is singular because leg pairs return to the ground at exactly the same time as other leg pairs take off. For example, leg pair 5 and leg pair 3 have this relationship.

5.2.6.3 An Example with Compound Leg Steps ($\sigma = 0.5$, $\omega = n$)

Figure 5.5 shows the gait diagram for a motion program that is just like the previous example, with the exception that it uses compound leg steps (as described in Section 5.1.6) instead of simple leg steps. The motion program depicted in this gait diagram is very similar to that used by caterpillars because it uses both a $\sigma = 0.5$ gait and because it includes forward pitching motion before lifting its feet off the ground and after placing its feet back down. While the gray bars in the figure indicate that a leg pair is performing a tilting motion, they also indicate that its feet are still on the ground supporting weight. These pitching movements complicate motion programs by causing four leg pairs to be in motion simultaneously. But the relative sequence of the actual foot lift-off and foot placement events has not changed between the simple step and compound step examples. Thus, while the leg pair trajectories of the motion program are more complicated, the basic *gait* remains the same as that shown in Fig. 5.4.

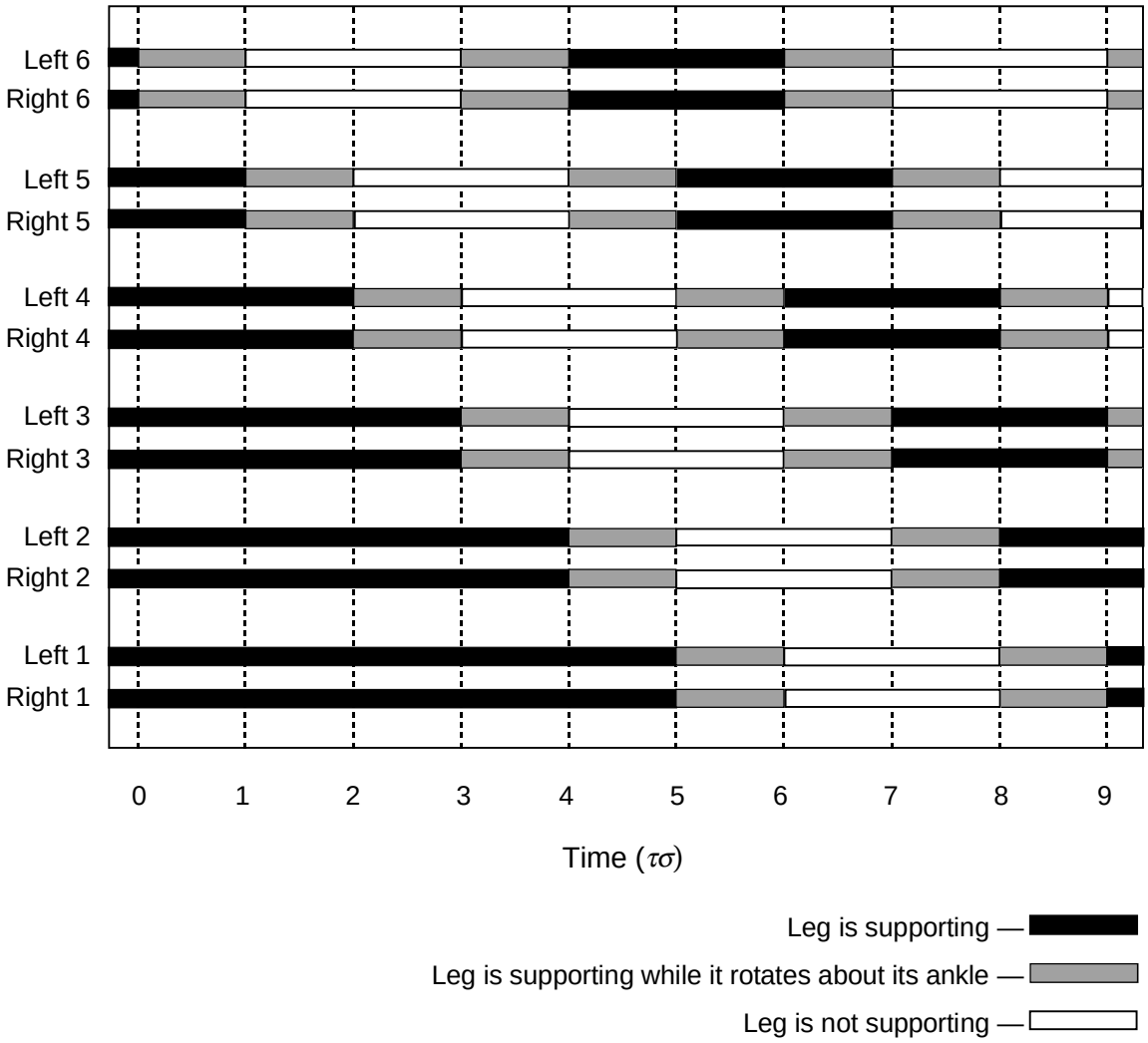

Figure 5.5 – Gait Diagram for a $\sigma = 0.5, \omega = 6.0$ Gait with Pitching, Performed on a 6 Leg Pair Robot.

5.2.6.4 An Example of Multiwave Locomotion ($\sigma = 1.0$, $\omega < n$)

The previous examples have all had wave intervals equal to their number of leg pairs. In this example we look at what happens when the wave interval is less than the number of leg pairs.

The gait diagram in Fig. 5.6 below is the same as that in Fig. 5.3, except that it has a shorter wave interval (ω) of 3.125. Using Eq. 5.11 to compute the time delay between waves, we get:

$$\delta t_w = \tau\sigma\omega = 3.125\tau\sigma \tag{5.16}$$

Looking at Fig. 5.6, we see that leg pair 4 starts its second step at 3.125 (since, once again, the graph is plotted with time units of $\tau\sigma$).

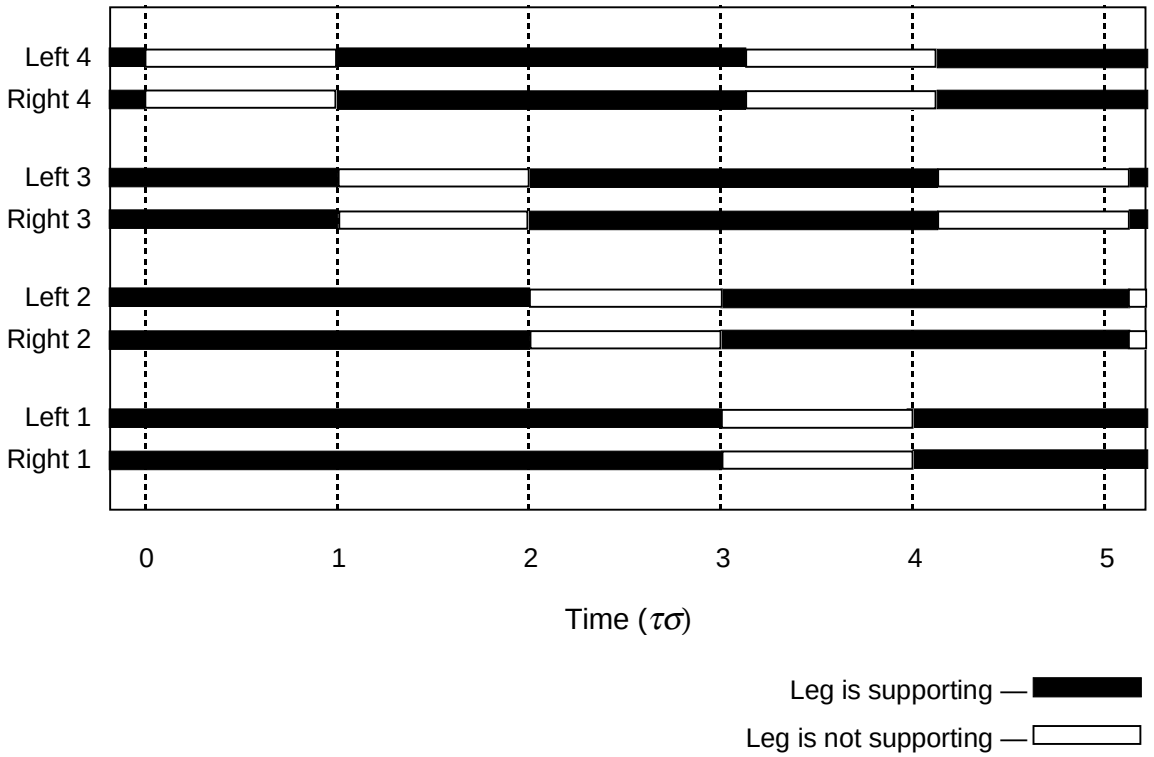

Figure 5.6 – Gait Diagram of a $\sigma = 1.0, \omega = 3.125$ Wave Gait, Performed on a 4 Leg Pair Robot.

Comparing Fig. 5.6 with Fig. 5.3, notice that, unlike the earlier $\omega = 4$ example, the $\omega = 3.125$ example has two active waves of motion on the robot for a portion of

each locomotion cycle. This example is similar to the "multiwave" locomotion observed with the caterpillars.

If the examples in Fig. 5.3 and Fig. 5.6 are assumed to be equivalent in all respects except the wave interval, we can compare their gait speeds using Eq. 5.15 as follows:

$$\frac{s_{cg}(5.3)}{s_{cg}(5.6)} = \frac{\frac{\lambda}{\tau\sigma\omega_{5.3}}}{\frac{\lambda}{\tau\sigma\omega_{5.6}}} = \frac{\omega_{5.6}}{\omega_{5.3}} = \frac{3.125}{4.000} = 0.7813 \qquad (5.17)$$

Thus, if all other things are equal, the $\omega = 3.125$ example is almost 22% faster than the $\omega = 4$ example.

5.2.6.5 An Example of Discrete Wave Locomotion ($\sigma = 0.6$, $\omega > n$)

In this example, a 6 leg pair robot is programmed to travel using overlapping compound leg steps and a wave interval, ω, that is *greater* than the number of its leg pairs. Specifically, the parameters that were used to create this example are a wave interval (ω) of 9, a step interval (σ) of 0.6, a transfer phase time (τ) of 1.0 second, a beginning pitch time (t_{pb}) of 0.4τ, and an ending pitch time (t_{pe}) of 0.4τ.

As explained in Section 5.2.3, when $\omega > n$, the locomotion cycles of a motion program may be discrete in the gait sense and sometimes in the general motion sense as well. The gait diagram in Fig. 5.7 will be used to illustrate what is meant by a discrete locomotion cycle in both senses.

Recalling that the wave interval is enumerated in terms of step intervals yields a way of visualizing the delay intervals in discrete motion programs. Specifically, each discrete locomotion cycle is separated from the others by an interval equal to the interval that would have existed if the robot had as many leg pairs as the wave

interval number. Accordingly, the difference between the wave interval and the actual number of leg pairs can be thought of as "phantom" leg pairs. Hence, in this example, the delay between locomotion cycles is as if there were 9 – 6 = 3 extra "phantom" leg pairs that executed leg steps that began at $t = 6\tau\sigma$, $t = 7\tau\sigma$, and $t = 8\tau\sigma$ respectively (see Fig. 5.7).

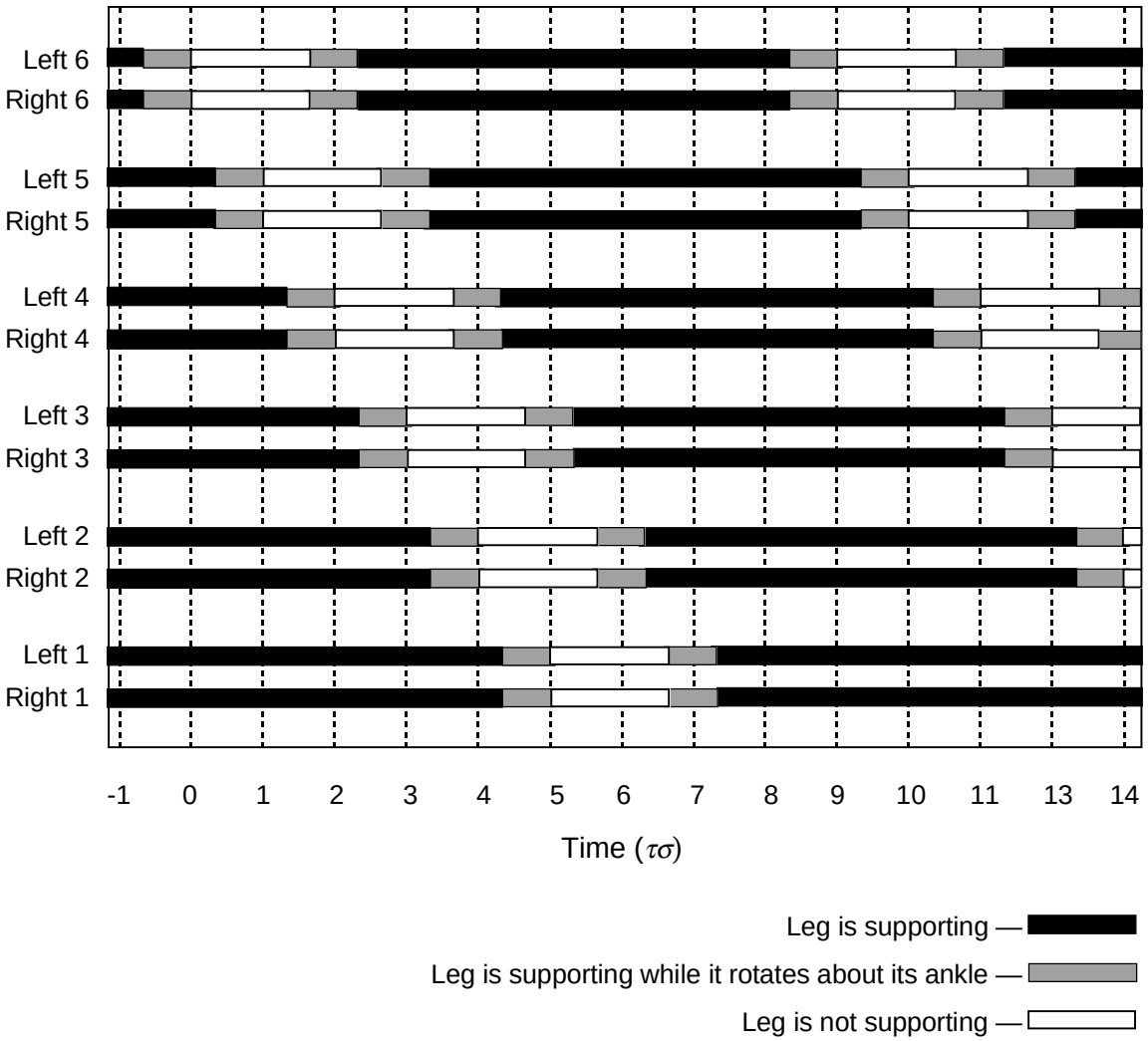

Figure 5.7 – Gait Diagram of a $\sigma = 0.6$, $\omega = 9$ Motion Program, Performed on a 6 Leg Pair Robot.

A motion program can be tested for discreteness in the gait sense by comparing the wave interval with the result of Eq. 5.8. By "gait sense" it is meant that this test evaluates the discreteness of the locomotion cycles based only upon the sequence of lifting and placing events of the feet and not on any pitching motions. Applying this test to the current example yields:

$$\omega_d = n + \frac{(1-\sigma)}{\sigma} = 6 + \frac{(1-0.6)}{0.6} = 6\tfrac{2}{3} \tag{5.18}$$

Since $\omega > \omega_d$ (i.e. $9 > 6\frac{2}{3}$), in this case, the motion program is discrete in the gait sense. Using Eq. 5.12 we can compute the time delay between the last placing event of one locomotion cycle and the first lifting event of the next locomotion cycle.

$$t_d = (\omega - \omega_d)\tau\sigma = (9 - 6\tfrac{2}{3})\tau\sigma = 2\tfrac{1}{3}\tau\sigma = 1.4 \text{ s} \tag{5.19}$$

This time delay is graphically illustrated in Fig. 5.7. Specifically, the first locomotion cycle ends when leg pair 1 returns its feet to the ground at $t = 6\frac{2}{3}\tau\sigma$ and the next locomotion cycle begins when leg pair 6 lifts its feet off the ground at $t = 9\tau\sigma$, a difference of $2\frac{1}{3}\tau\sigma$, or 1.4 seconds, as computed in Eq. 5.19.

Looking at the motion program in Fig. 5.7 from the point of view of general motion (including any pitching motions of leg pairs that are on the ground), we can determine whether it is discrete by applying Eq. 5.9 and comparing the result with the wave interval.

$$\omega_d = n + \frac{(1-\sigma)}{\sigma} + \frac{t_{pb} + t_{pe}}{\tau\sigma} = 6 + \frac{(1-0.6)}{0.6} + \frac{0.4\tau + 0.4\tau}{0.6\tau} = 8 \tag{5.20}$$

Since $\omega > \omega_d$ (i.e. 9 > 8) we see that the motion program is also discrete in the "motion sense". The time delay, between the cessation of all motion of one locomotion cycle and the initiation of motion of the next locomotion cycle, can be calculated by applying Eq. 5.12 as shown below.

$$t_d = (\omega - \omega_d)\tau\sigma = (9-8)\tau\sigma = \tau\sigma = 0.6\text{s} \tag{5.21}$$

Looking at Fig. 5.7, this computed delay is apparent as the pitching motion of leg pair 1 stops at $t = 7\frac{1}{3}\tau\sigma$ and then leg pair 6 starts its pitching motion at $t = 8\frac{1}{3}\tau\sigma$. The difference between these values is $\tau\sigma = 0.6\text{s}$, which is in agreement with the result of Eq. 5.21 above.

A motion program that is discrete in the motion sense is also always discrete in the gait sense, assuming that the "on the ground" motions are limited to what has been described here as compound leg steps. However, a motion program can be discrete in the gait sense without being discrete in the motion sense.

Also note that while it is an integer in this example, ω is not required to be an integer and that the general-case speed equation (Eq. 5.15) works for any value of $\omega \geq 1/\sigma$, including all discrete cases.

5.3 Motion Programming Guidelines

Having presented the essentials of the motion programming method developed for the multibody passive-legged crawling vehicle, we now consider some preliminary guidelines for motion programming. These guidelines include a discussion of the practical limits on crawling vehicle velocity and recommendations for varying vehicle velocity, selecting gait parameters, and constraining when gait parameters may be modified at runtime. These are stated as "preliminary" because use of the simulation tools to be discussed in Ch. 6 and Ch. 7 will almost certainly further illuminate these issues.

5.3.1 Practical Limitations on Vehicle Velocity

According to Eq. 5.15, it is theoretically possible to specify an arbitrarily small value for τ to produce higher vehicle speeds. However, as a practical matter, the value of τ (and hence vehicle speed) is limited by the available power, torque, and peak speed of the actuators.

Equation 5.15 also shows that the vehicle velocity is inversely related to both σ and ω. For example, with the wave interval, a case of $\omega = n$ corresponds to the robot performing exactly one wave of leg step motion at a time. Similarly, $\omega = \frac{1}{2}n$ specifies that two waves of motion will execute simultaneously, thus doubling the speed of the vehicle, assuming that all other things are the same as in the $\omega = n$ example. However, the values of the step interval and wave interval that can be safely used are limited because some combinations σ and ω will not be stable.

Finally, it may be unsafe to travel at higher speeds if the path planning computation speed (including processing terrain sensing and mapping data) cannot keep up.

5.3.2 Modulating Vehicle Velocity

When operating a crawling vehicle, there are many possible scenarios where it would be desirable to vary its speed. Looking at Eq. 5.15, we see that the speed of the vehicle can be increased by increasing the stride length, λ, reducing the transfer phase time, τ, reducing the step interval, σ, or reducing the wave interval, ω. Using the stride length to vary the speed of the vehicle is a poor choice because, for efficiency and longitudinal stability, the stride length used will generally be the longest one that can be used safely. As discussed in the previous section, the choices of which step intervals and wave intervals to use are often constrained by the stability requirements of the vehicle. Consequently, the selection of σ and ω will frequently be determined by the difficulty of the terrain that the robot is traversing

and by the number of leg pairs in the robot. Thus, modulating the transfer phase time, τ, will usually be the most convenient method of varying vehicle speed.

Speed transitions, such as increasing the wave speed of one wave but not speeding up the prior wave, can cause difficulties because the following wave may overtake the prior one. Effectively, such approaches are making τ non-uniform in its application along the length of the robot, and are also varying the values of both τ and ω in time. That is, the transfer phase time of one wave will be different from the next wave, and as the waves propagate up the length of the robot's body, their wave speeds will vary, as will the wave interval between them. While, with careful planning and coordination, this could be made to work in some cases, it is much simpler to constrain all simultaneously active waves to have the same value of τ. Wave acceleration can still be accomplished by smoothly reducing τ simultaneously for all the active leg steps. If the path update frequency of the controller is constant, then shortening τ corresponds to increasing the parametric variable spacing between successive via points. In such manner the vehicle can smoothly accelerate without the possibility of having waves collide. Similarly, the robot can decelerate by smoothly increasing the transfer phase time for all the waves.

Thus, this idea of instantaneously uniform transfer phase time (that the same τ applies to the entire robot at any one instant in time, but that τ can be varied in time) can be used to modulate the speed of the vehicle without the danger of having waves interfere with each other, causing assembly failures. In view of the simplicity and effectiveness of this method, there seems to be little benefit in deriving the numerous constraint equations and motion planning heuristics that would be necessary to use non-uniform transfer phase time/non-constant wave interval approaches.

5.3.3 Selecting Gait Parameters

Based upon the discussions in Sections 5.3.1 and 5.3.2, the following are some guidelines for selecting gait parameters:

The minimum value of the wave interval, ω, is constrained by the stability requirements of the vehicle. However, when traversing easy terrain, ω can be modulated to trade-off between stability and speed. For example, reducing the value of ω will cause additional waves of leg steps to simultaneously execute on the vehicle, thus increasing its velocity.

The minimum value of the step interval, σ, is also constrained by vehicle stability, but to a lesser extent than ω (i.e. it is only constrained by stability when the robot is operating on the roughest terrain). In general, σ should be selected for efficiency. The more efficient gaits also tend to produce faster locomotion. Therefore, it is wasteful to use σ to modulate vehicle velocity.

The minimum transfer phase time, τ, is constrained by the dynamics of the robot, including considerations such as the peak power required, peak actuator velocity, and slope of the terrain underfoot, as well as others. Specifically, the dynamic state of the robot at any point in time determines the minimum value of τ, which, in turn, limits the maximum vehicle velocity. From this minimum value, τ can be increased so as to reduce the velocity of the vehicle. Thus, τ is chosen primarily based upon the desired vehicle velocity.

5.3.4 Constraints on Gait Parameter Modification

Section 5.3.2 described some potential pitfalls associated with the choice of motion programming parameters. Some recommendations for avoiding these problems

when programming the motion of wave gaits for the crawling vehicle are summarized below:

- Uniform transfer phase time — The same value of τ applies to the entire robot at any one instant in time. All simultaneously active waves should use the same τ. However, this instantaneously uniform τ need not be constant, it can be varied in time in order to produce accelerations and decelerations of the vehicle.

- Constant step interval — Once a wave has begun using a certain value of σ, it should continue to use the same step interval as it propagates up the length of the robot's body. However, the choice of step interval does not have to be uniform from one wave to the next; each wave is independent. As will be demonstrated via simulation in Ch. 7, in order for the robot to assemble, different step intervals result in different constraints on stride lengths. Thus, a mid-locomotion cycle transition from one step interval to another can result in assembly problems and/or a situation in which leg pairs are unevenly spaced along the length of the robot once the wave has passed. While not a disaster, uneven leg pair spacing reduces both the longitudinal stability margin and the maximum speed, as it may take several locomotion cycles for the vehicle to re-extend the leg pair spacing back to normal.

- Constant wave interval — Once a wave begins, defining the wave interval, ω, between it and the previous wave, the two waves should maintain that same wave interval as they propagate up the length of the robot's body. However, wave intervals are independent and do not have to be uniform. Thus, for example, the wave interval between the first wave and the second wave can be different than the wave interval between the second wave and the third wave, but once a particular wave interval has begun, it should remain constant as it moves along the robot's length.

In practice, obeying the latter two "rules" is easy, so long as changes to σ and ω are only applied to new waves as they are initiated when the rearmost leg pair begins its stride.

Motion programs that do not follow these guidelines can result in non-periodic gaits that cannot be sustained for more than a few locomotion cycles before assembly failures result. While it is possible to compensate by varying stride lengths and other parameters to avoid assembly failures, each new permutation of such gaits would require custom-designed motion programs for *each* locomotion cycle. Thus, following these motion programming rules is not absolutely essential, but if they are not followed, the motion programming will dramatically increase in complexity, causing each motion programming task to become a separate research problem.

5.4 Chapter Summary

This chapter has presented a method of motion programming for the multibody passive-legged crawling vehicle, discussing the major issues of specifying both leg pair trajectories and gaits. For leg pair trajectory specification, the final method settled upon is to use parametric cubic curves to specify both *xyz* translations and roll-yaw-pitch rotations for each leg pair. Timing issues relating to gaits are specified by defining two parameters, the step interval and the wave interval. Equations relating these new parameters to vehicle performance were presented, as well as several representative gait examples. Finally, a number of recommendations for effective motion programming were offered.

The next chapter will provide a brief overview of the simulation software developed to embody these motion programming concepts and combine them with the modeling techniques presented in Ch. 4.