Evolution of Walking
Walking

Intelligence as Adaptive Behavior
Perspectives in Artificial Intelligence
An Experiment in Computational Neuroethology

Randall D. Beer
Leg Model

States
- Leg Angle: $\phi$
- Leg Angular Velocity: $\omega = \dot{\phi}$
- Joint-to-Foot Distance: $d$

Inputs
- Backward Swing Effector: BS
- Forward Swing Effector: FS
- Foot Effector: FT

Outputs
- Leg Angle Sensor: AS
Foot is up when $FT \leq 0.5$

Resultant torque on leg

$MaxLegTorque \ast (FS - BS)$

Integrate $\phi$ and $\omega$

$\ddot{\phi} = MaxLeqTorque \ast (FS - BS)$
Stance Dynamics

Foot is down when $FT > 0.5$

When foot goes down  \[ d = LegLength \times \cos \phi \]

Resultant force on body \[ MaxLegForce \times (BS - FS) \]

Convert to foot-centered rectangular coordinates
\[
\begin{align*}
\dot{x} &= -d \tan \phi \\
\dot{v} &= \frac{d}{dt} (-d \tan \phi) = -\omega d \sec^2 \phi
\end{align*}
\]

Integrate $x$ and $v$: \[ \ddot{x} = MaxLegForce \times (BS - FS) \]

Convert back to angular coordinates
\[
\begin{align*}
\phi &= \tan^{-1} \left( \frac{-x}{d} \right) \\
\omega &= \frac{d}{dt} \left( \tan^{-1} \left( \frac{-x}{d} \right) \right) = \frac{-vd}{d^2 + x^2}
\end{align*}
\]
Model Configuration

LegLength = 15
MaxLegTorque = 1/40
MaxLegForce = 1/20

$d = 15 \cos \phi$

$-d \tan \phi$

$\phi, \omega$

$\pi/6$

Absolute Forward Limit

Absolute Backward Limit

Forward Angle Limit

Backward Angle Limit

Support

Swing

Stance Power

Snap Back
// Step the insect using a general CTRNN

void LeggedAgent::Step(double StepSize)
{
    double force = 0.0;

    // Update the leg angle sensor
    // NervousSystem.SetNeuronExternalInput(1, Leg.Angle * 5.0/ForwardAngleLimit);
    // Update the nervous system
    NervousSystem.EulerStep(StepSize);
    // Update the leg effectors
    if (NervousSystem.NeuronOutput(1) > 0.5) {Leg.FootState = 1; Leg.Omega = 0;}
    else Leg.FootState = 0;
    // Compute the force applied to the body
    double f = Leg.ForwardForce - Leg.BackwardForce;
    // Update the position of the body
    vx = vx + StepSize * force;
    if (vx < -MaxVelocity) vx = -MaxVelocity;
    if (vx > MaxVelocity) vx = MaxVelocity;
    cx = cx + StepSize * vx;
    // Update the leg geometry
    Leg.JointX = Leg.JointX + StepSize * vx;
    if (Leg.FootState == 1.0) {
        Leg.Omega = (angle - Leg.Angle)/StepSize;
        Leg.Angle = angle;
    } else {
        vx = 0.0;
        Leg.Omega = Leg.Omega + StepSize * MaxTorque * (Leg.BackwardForce - Leg.ForwardForce);
        if (Leg.Omega < -MaxOmega) Leg.Omega = -MaxOmega;
        if (Leg.Omega > MaxOmega) Leg.Omega = MaxOmega;
        Leg.Angle = Leg.Angle + StepSize * Leg.Omega;
        if (Leg.Angle < BackwardAngleLimit) {Leg.Angle = BackwardAngleLimit; Leg.Omega = 0;}
        if (Leg.Angle > ForwardAngleLimit) {Leg.Angle = ForwardAngleLimit; Leg.Omega = 0;}
        Leg.FootX = Leg.JointX + LegLength * sin(Leg.Angle);
        Leg.FootY = Leg.JointY + LegLength * cos(Leg.Angle);
    }
    // If the foot is too far back, the body becomes "unstable" and forward motion ceases
    if (cx - Leg.FootX > 20) vx = 0.0;
}
An Example Walk
Initialization

$Head = \{(29.4, 0), (21.9, 11.25), (14.4, 0), (21.9, -11.25), (29.4, 0)\};
$Torso = \{(18, 6), (0, 12), (-30, 5), (-30, -9), (0, -12), (18, -6)\};
$LeftAnt = \{(25.65, -5.6), (65, -30)\}; $RightAnt = \{(25.65, 5.6), (65, 30)\};
$LeftCercus = \{(-30, -2), (-34, -8)\};
$RightCercus = \{(-30, 2), (-34, 8)\};

DisplayBody[\{JointX, JointY, FootX, FootY, FootState\}] :=
  Module[\{Head, Torso, LeftAnt, RightAnt, LeftCercus, RightCercus, Leg, Foot\},
    Head = Map[\{#1 + JointX, #2\} &, $Head];
    Torso = Map[\{#1 + JointX, #2\} &, $Torso];
    LeftAnt = Map[\{#1 + JointX, #2\} &, $LeftAnt];
    RightAnt = Map[\{#1 + JointX, #2\} &, $RightAnt];
    LeftCercus = Map[\{#1 + JointX, #2\} &, $LeftCercus];
    RightCercus = Map[\{#1 + JointX, #2\} &, $RightCercus];
    Leg = \{(JointX, JointY), \{FootX, FootY\}\};
    Foot = \{FootX, FootY\};
    Show[Graphics[
      \{RGBColor[1, 0.6, 0], Polygon[Head], Polygon[Torso]},
      \{RGBColor[0.6, 0.4, 0.2], Line[LeftAnt], Line[RightAnt], Line[LeftCercus],
        Line[RightCercus], Thickness[0.005], Line[Head], Line[Torso]},
      \{RGBColor[0.6, 0.4, 0.2], Thickness[0.005], Line[Leg]},
      If[FootState == 1, \{RGBColor[1, 0, 0], PointSize[0.02], Point[Foot]\}, {}],
      AspectRatio -> 1/3.5, PlotRange -> \{(-35, 210), (-35, 35)\}, Frame -> True,
      FrameTicks -> \{False, False\}]]

Animation

T = Import["/Users/rdbeer/Desktop/out.dat"];
Animate[DisplayBody[T[[i]]], \{i, 1, Length[T], 1\}]
Evolving Walking

15-35 Free Parameters
(18-40 with Sensor Weights)

\[ I_i = w_i \phi \]
Three Different Conditions

- Sensory feedback *never* available
- Sensory feedback *always* available
- Sensory feedback *intermittently* available
Central Pattern Generator

Velocity

Foot

Backward Swing

Forward Swing

Interneuron A

Interneuron B
Reflexive Pattern Generator

With Sensory Feedback

Without Sensory Feedback
Mixed Pattern Generator

With Sensory Feedback

Without Sensory Feedback
Entrainment in Mixed Pattern Generators

Low Frequency Drive

High Frequency Drive
Six-Legged Walking

Angle Sensor

Foot
Backward Swing
Forward Swing

Angle Sensor

Foot
Backward Swing
Forward Swing

Angle Sensor

Foot
Backward Swing
Forward Swing

Angle Sensor

Foot
Backward Swing
Forward Swing

Angle Sensor

Foot
Backward Swing
Forward Swing

Angle Sensor

Foot
Backward Swing
Forward Swing

Angle Sensor

Foot
Backward Swing
Forward Swing
Six-Legged Results

RPG

CPG

MPG
Evolutionary Implications of Center-Crossing CTRNNs

**Graph 1:** Evolutionary performance over mutation variance for different seeding methods. The graph compares the mean best performance (y-axis) across generations (x-axis) for different mutation variances. The lines represent different seeding conditions: seeded with random and seeded with random CC. The graph shows a clear trend of improved performance with increasing mutation variance for both seeding methods.

**Graph 2:** Evolutionary performance over generations for different levels of variance. The graph compares the mean best performance (y-axis) across generations (x-axis) for different levels of variance. The lines represent low and high variance conditions, with high variance showing a slower but consistent improvement in performance.

Seeded with Random
Seeded with Random CC
Low $\sigma^2$
High $\sigma^2$