Roombots-CPG, Symmetries and online optimization.

Tuleu Alexandre

Midterm presentation

November 3, 2008
Goal of the project.

- Implement several Roombot configurations, and find a way to identify morphological symmetries, in order to design a CPG along this structure.
Goal of the project.

- Implement several Roombot configurations, and find a way to identify morphological symmetries, in order to design a CPG along this structure.
- Check out and test new optimization methods.
Goal of the project.

- Implement several Roombot configurations, and find a way to identify morphological symmetries, in order to design a CPG along this structure.
- Check out and test new optimization methods.
- Test and select different CPG type.
Goal of the project.

- Implement several Roombot configurations, and find a way to identify morphological symmetries, in order to design a CPG along this structure.
- Check out and test new optimization methods.
- Test and select different CPG type.
  - add sensor feedback from the roombot to the CPG.
Goal of the project.

- Implement several Roombot configurations, and find a way to identify morphological symmetries, in order to design a CPG along this structure.
- Check out and test new optimization methods.
- Test and select different CPG type.
  - add sensor feedback from the roombot to the CPG.
- Lot of previous work done by former student (YaMor host 3.0).
A Way to Characterize a Roombot Structure.

Based on previous work on YaMor [?].
A Way to Characterize a Roombot Structure.

- Based on previous work on YaMor [?].
- We represent the structure as a graph:
  - Each module is a vertex.
A Way to Characterize a Roombot Structure.

Based on previous work on YaMor [?].
We represent the structure as a graph:
  - Each module is a vertex.
  - Each connection between modules is an edge.

Different from Jocelyne's one, since our graph must be servo values independent.
A Way to Characterize a Roombot Structure.

- Based on previous work on YaMor [?].
- We represent the structure as a graph:
  - Each module is a vertex.
  - Each connection between modules are an edge.
  - We put label on edge that fully characterize the connection between modules.

Tuleu Alexandre
Roombots-CPG, Symmetries and online optimization.
A Way to Characterize a Roombot Structure.

Based on previous work on YaMor [?]. We represent the structure as a graph:
- Each module is a vertex.
- Each connection between modules is an edge.
- We put labels on edges that fully characterize the connection between modules.

Different from Jocelyne's one, since our graph must be servo values independent.
A Way to Characterize a Roombot Structure.

Based on previous work on YaMor [?].

We represent the structure as a graph:

- Each module is a vertex.
- Each connection between modules is an edge.
- We put labels on edges that fully characterize the connection between modules.

Different from Jocelyne’s one, since our graph must be servo values independent.
Main Idea: Symmetries in a CPG, conduct to the same symmetries on the gait.
Main Idea: Symmetries in a CPG, conduct to the same symmetries on the gait.

Find a way to detect symmetries in a structure, and then determine the available gait.
Main Idea: Symmetries in a CPG, conduct to the same symmetries on the gait.

Find a way to detect symmetries in a structure, and then determine the available gait.

Maybe try to memorize some patterns.
Our goal is to learn a gait by optimization:

\[
\max_{\mathbf{X}} f(\mathbf{X}, \alpha)
\]

We don’t know a mathematical expression for \( f \), but \( f \) is:

- nonlinear.
- multimodal.
Lot of random, and evolutionary algorithms:
Lot of *random*, and *evolutaniory* algorithms:
PSO,
Lot of random, and evolutionary algorithms: PSO, Cross-Entropy Method,
Lot of *random*, and *evolutaniory* algorithms: PSO, Cross-Entropy Method, Harmony search.
Different Algorithm for Online Learning.

- Lot of *random*, and *evolutionary* algorithms: PSO, Cross-Entropy Method, Harmony search.
- Powel’s method can be extended to multimodal function:
Lot of random, and evolutionary algorithms: PSO, Cross-Entropy Method, Harmony search.
Powel’s method can be extended to multimodal function:
   use of systematical search instead of golden section search.
Different Algorithm for Online Learning.

- Lot of *random*, and *evolutaniory* algorithms: PSO, Cross-Entropy Method, Harmony search.
- Powel’s method can be extended to multimodal function:
  - use of systematical search instead of golden section search.
  - Boender-Rinnooy-Stougie-Timmer (BRST) algorithm.
What is more important than the algorithm is the expression of objective function.
Estimating the objective function.

- What is more important than the algorithm is the expression of objective function.
- We want to avoid "traps".

\[
\text{f}(X) = s(t_{\text{end}}) - s(t_{\text{deb}}) - K \int_{t_{\text{deb}}}^{t_{\text{end}}} + T (\dot{s}(t) - \dot{s}(t - T))^2 \, dt (2)
\]
What is more important than the algorithm is the expression of objective function.

- We want to avoid "traps".
  - done by integrating over more time, but has a cost.
What is more important than the algorithm is the expression of objective function.

We want to avoid "traps".

- done by integrating over more time, but has a cost.

Idea: measure the periodicity of the function:

\[
\begin{align*}
  f(X) &= s(t_{\text{end}}) - s(t_{\text{deb}}) \\
  &\quad + \int_{t_{\text{deb}}}^{t_{\text{end}}} (\dot{s}(t) - \dot{s}(t - T))^2 dt
\end{align*}
\]
What is more important than the algorithm is the expression of objective function. We want to avoid "traps". done by integrating over more time, but has a cost. Idea : mesure the periodicity of the function :

\[ f(X) = \frac{s(t_{\text{end}}) - s(t_{\text{deb}})}{t_{\text{end}} - t_{\text{deb}}} - K \int_{t_{\text{deb}}+T}^{t_{\text{end}}} (\dot{s}(t) - \dot{s}(t - T))^2 \, dt \] (2)
Modelisation of the contacts.

Now modelized by a sphere.
Modelisation of the contacts.

- Now modelized by a sphere.
- The real shape collision with a plane isn’t so hard to compute.
Modelisation of the contacts.

- Now modelized by a sphere.
- The real shape collision with a plane isn’t so hard to compute.

Problem:
Difficulty to implement it in webots.
1. We project the center of the module on the plane.
1. We project the center of the module on the plane.

2. We detect if we are in one of the three cones or in other half-space.
Detection collision with a plane.

1. We project the center of the module on the plane.
2. We detect if we are in one of the three cones or in other half-space.
3. If in the cone we calculate the deepest and highest point of the face circle.
Detection collision with a plane.

1. We project the center of the module on the plane.
2. We detect if we are in one of the three cones or in other half-space.
3. If in the cone we calculate the deepest and highest point of the face circle.
   1. If the highest is under the plane, we give back three point on the circle.
1. We project the center of the module on the plane.

2. We detect if we are in one of the three cones or in other half-space.

3. If in the cone we calculate the deepest and highest point of the face circle.
   
   1. If the highest is under the plane, we give back three points on the circle.
   
   2. If the highest is over the plane, then we just send back the deepest point of the circle.
Mainly have coded and debugged most of my framework.
Mainly have coded and debugged most of my framework.

Attempt some first result with Powell algorithm and simple CPG this week.
Questions ?