Programmable Central Pattern Generators

Florian Hackenberger
florian@hackenberger.at

University of Technology, Graz

November 14, 2006
Introduction

What is a Central Pattern Generator?
How can we program such a thing?
Synchronisation

PCPG Theory

A simple programmable Hopf oscillator
Coupling several oscillators to create a PCPG
Maintaining a phase offset

Applications

Biped locomotion control

Summary
Biological phenomenon

- Neural Network which generates multidimensional rhythmic signals
- Found in vertebrate as well as invertebrate animals
- Used for locomotion control (among other things)
Mathematical Model

- System of coupled oscillators
- Each oscillator has a stable limit cycle
- Output is a weighted sum over the output of the individual oscillators
Programming Central Pattern Generators

Each oscillator has state variables which can adapt to:

- The frequency $\omega$ of an input signal
- The amplitude $\alpha$ of an input signal
- The phase offset $\phi_{diff}$ from the first oscillator of the system

The frequency and amplitude variables are *frozen* if the input signal vanishes.
Why do we need Synchronisation?

- It is the reason why frequency learning works at all
- If the phase $\phi$ of a single oscillator is perturbed the system produces quite a different output
- The amplitude of an oscillator is stable, but the phase is free
- Synchronisation can make the PCPG resistant to such perturbations by maintaining internal coherence
How does Synchronisation work in general?

- Introduces a *weak* coupling between two or more oscillators, which acts on the phase $\phi$
- Leads to phase locking
- Example: Pendulum clocks on a common support
A simple programmable Hopf oscillator

Coupling several oscillators to create a PCPG

Maintaining a phase offset

Equation System

\[ \dot{r} = (\mu - r^2) r + \epsilon F \cos(\phi) \]
\[ \dot{\phi} = \omega - \frac{\epsilon}{r} F \sin(\phi) \]
\[ \dot{\omega} = -\epsilon F \sin(\phi) \]

Describes an oscillator with radius \( \sqrt{\mu} \) (possibly perturbed) and frequency \( \omega \) (which adapts to \( F \))
Learning $\omega$ of $\cos(30t)$

\[
\begin{align*}
\dot{r} &= (\mu - r^2) r + \epsilon F \cos(\phi); \\
\dot{\phi} &= \omega - \frac{\epsilon}{r} F \sin(\phi); \\
\dot{\omega} &= -\epsilon F \sin(\phi)
\end{align*}
\]

If the oscillator begins to synchronise with a frequency component of the external signal, the frequency is adapted relative to the coupling $-\frac{\epsilon}{r} F \sin(\phi)$, which is applied to the phase. This speeds up the synchronisation. As soon as the two signals are completely synchronised, the frequency $\omega$ converges and finally remains constant.
The signal $F$ is redefined as $F(t) = P_{teach} - \alpha \cos(\phi)$ and the state variable $\alpha$ increases if the frequency of the oscillator correlates with one frequency component of the input signal. As $\alpha$ increases, the component disappears from the input signal. The redefined $F$ is a kind of error signal, which tends to zero as soon as convergence sets in (see next slide).
Learning $\omega$ and $\alpha$ of $\cos(30t)$

$\alpha$ starts to converge as soon as $\omega$ has been learnt.

Frequency and amplitude states

The error of input vs. output ($F$)
The output for the polar coordinate version of the oscillator is in fact

\[ \sum_{i=0}^{N} \alpha_i R_i \cos(\phi_i) \]
A simple programmable Hopf oscillator
Coupling several oscillators to create a PCPG
Maintaining a phase offset

Modified differential equation system

A system of $i$ coupled (via the additive term
$\tau \sin(R_i - \phi_i - \phi_{i,\text{diff}})$) adaptive Hopf oscillators.

\[
\begin{align*}
\dot{r}_i &= \gamma(\mu - r_i^2)r_i + \epsilon F \cos(\phi_i) \\
\dot{\phi}_0 &= \omega_i - \frac{\epsilon}{r_i} F \sin(\phi_i) \\
\dot{\phi}_i &= \omega_i - \frac{\epsilon}{r_i} F \sin(\phi_i) + \tau \sin(R_i - \phi_i - \phi_{i,\text{diff}}); \ \forall i > 0 \\
\dot{\omega}_i &= -\epsilon F \sin(\phi_i) \\
\dot{\alpha}_i &= \eta F \cos(\phi_i) r_i \\
\dot{\phi}_{i,\text{diff}} &= \lambda \sin(R_i - \phi_i - \phi_{i,\text{diff}}); \ \forall i > 0 \\
R_i &= \frac{\omega_i}{\omega_0} \phi_0
\end{align*}
\]
Learning a $\phi$ offset

Two phase signals, $\phi_0$ with $\omega_0 = 1$ and $\phi_1$ with $\omega_1 = 2$ and a $\phi$ shift of 0.4.

The slow phase $\phi_0$, the fast phase $\phi_1$ and $R_1$

phi offset is $\sin(R_i - \phi_i)$
Two phase signals, $\phi_0$ with $\omega_0 = 1$ and $\phi_1$ with $\omega_1 = 2.1$ and a $\phi$ shift of 0.3. The coupling compensates for the 0.1 deviation from an integer frequency relationship between the oscillators. As soon as the coupling is $\neq 0$ the phase of the second oscillator is accelerated, bringing the phase difference back to $\phi_{i,diff}$ (not shown).
One CPG for each joint. Coupling to maintain synchronisation.
Modified equations

We add a coupling term to the first oscillator of each joint (apart from the two hip joints) and we add eight state variables \((\phi_k, \text{diff}ext)\) which are supposed to converge to the phase difference between the joints.

\[
\begin{align*}
\dot{\phi}_{0,k} &= \omega_{0,k} - \frac{\epsilon}{r_{0,k}} F \sin(\phi_{0,k}) + \\
&\quad \tau_{\text{ext}} \sin(R_{0,k} - \phi_{0,k} - \phi_{k,\text{diff}ext}); \quad \forall k \neq \{0, 5\} \\
\dot{\phi}_{k,\text{diff}ext} &= \lambda \sin(R_{0,k} - \phi_{0,k} - \phi_{k,\text{diff}ext}); \quad \forall k \neq \{0, 5\} \\
R_{0,k} &= \frac{\omega_{0,k-1}}{\omega_{0,k}} \phi_{0,k}
\end{align*}
\]
Learning the Fujitsu Hoap-2 walking gait
Summary

A PCPG is a system of coupled oscillators which act as a kind of function approximator

- **Advantages**
  - Learning embedded into the dynamics of the system (can be switched on/off online)
  - Stable limit cycle (perturbations don’t pose problems)
  - Smooth modulation of frequency and amplitude
  - Simple integration of feedback pathways

- **Disadvantages**
  - Quality of results depend heavily on the initial frequencies
Changes compared to L. Righetti’s approach

- Dynamical system rewritten in polar coordinates
  - Equations are easier when applying a coupling to the phase
- More Kuramoto like coupling scheme
  - Righetti’s synchronisation coupling acts on $\dot{x}$ only, therefore perturbing the circular output
  - The coupling presented here acts on the phase only, therefore the output of each oscillator is always a cycle.
- Slowly decreasing $\epsilon$ decreases the error after switching off the input signal
  - Using something like $\epsilon = \epsilon_{init} 0.9996^{t-t_{cooling}}$; $\forall t > t_{cooling}$
Future Work

- Port the Matlab implementation to C++ (leveraging an implementation of Righetti’s approach) in order to improve Hoap-2 walking
- Design a coupling with a spike train (useful for Hoap-2 walking, replacing the notion of phase resetting)
Please note that the approach described in the second article is a bit different from the approach presented here. The method shown in the article has severe problems regarding the convergence of the phase difference (see lowest graph of Fig. 4). The approach presented solves these deficiencies.

A. Pikovsky, R. Rosenblum and J. Kurths
Synchronization, A universal concept in nonlinear sciences
Cambridge Nonlinear Sciences Series, vol. 12, UK, 2001

L. Righetti, J. Buchli and A. Ijspeert
Dynamic Hebbian learning for adaptive frequency oscillators

L. Righetti, J. Buchli and A. Ijspeert
From Dynamic Hebbian Learning for Oscillators to Adaptive Central Pattern Generators
In *Proceedings of the 3rd International Symposium on Adaptive Motion in Animals and Machines*, Ilmenau, 2005