Predictive Gaze Stabilization During Periodic Locomotion Based On Adaptive Frequency Oscillators

Sébastien Gay1,2, Auke Ijspeert1 and José Santos Victor2
1Swiss Federal Institute of Technology, Lausanne, Switzerland, 2Instituto Superio Técnico, Lisbon, Portugal

Abstract—In this paper we present an approach to the problem of stabilizing the gaze of legged robots using Adaptive Frequency Oscillators to learn the frequency, phase and amplitude of the optical flow and generate compensatory commands during robot locomotion. Assuming periodic and nearly sine shaped motion of the head of the robot, the system successfully stabilizes the gaze of the robot, whether the robot itself is moving, or an external object is moving relative to the robot. We present experiments in simulation and, for object tracking, with a real robotics setup, the Hoap 3, showing that the system can be successfully applied to gaze stabilization during locomotion, even when the feedback loop is very slow and noisy.

I. INTRODUCTION

Vision is, for animals and robots, the most versatile sensor to provide information about the surrounding environment. However, vision is most efficient when the image (and thus the gaze) is stable since a moving gaze causes motion blur. Evolved animals use saccades when switching gaze direction to minimize the time during which the image is moving. During locomotion, compensatory movements of the eyes and head aim at minimizing the retina slip. The same issue is present when dealing with robots since most vision processing algorithms reach optimal performance with a stable image.

Head stabilization systems exist in the robotics literature, many of them being based on models of the vestibulo-ocular reflex observed in many vertebrates ([14]). These systems typically use a vestibular sensor (IMU, accelerometer etc.) as main sensory input to excite a leaky integrator. The remaining retinal slip (usually measured by optical flow) is then used to calibrate the gains of this integrator. Kawato’s Feedback-Error-Learning model ([4]) is applied to the gaze stabilization problem in [13] where it is extended with a nonparametric regression network to improve the optokinetic response. In [5], the authors implement the Recurrent Decorrelation Control model [2] which forms a recurrent network with an artificial brainstem getting as input rotational speeds from the vestibular sensor, and an artificial cerebellum getting input from the brainstem and the retinal slip, and feeding back its output to the brainstem. A single neural network is used in [8] and excited directly by both the vestibular sensor output and the optical flow from the camera image to estimate the optimal compensatory motor command.

These systems reach very good performance but rely highly on the availability of a fast (typically around 500Hz) vestibular sensor in the head of the robot. Although this kind of sensors becomes more accessible, many robots still do not have an IMU in the head of the robot, but rather in the trunk. Very few approaches tackle the problem of head stabilization specifically during locomotion. The work in [11] relies on a forward kinematics and genetic algorithm to build an internal model of the head motion and compensate for it using a feed-forward CPG based controller. This method however relies on offline optimization for the CPG parameters which has to be done for each different gait and is thus not very suitable for gaits changing in time (to cope with environmental specificities for instance).

In this paper we propose a system for stabilizing the head of a legged robot during locomotion, which only relies on optical flow information. Assuming a periodic movement of the head (as is usually the case for legged locomotion), the system uses Adaptive Frequency Oscillators to learn the frequency and phase shift of the optical flow and generate compensatory movements to minimize the head motion. At convergence, the system is mostly feedforward and the feedback signal (the optical flow) is only used to finely tune the parameters of the oscillator. The system further shows the same interesting properties in terms of control as other oscillators (smooth modulation of parameters, resistance to perturbations etc.). This system is efficient even when using relatively slow cameras (< 30Hz) and is predictive in the sense that unlike reactive systems which use the last few sensor values to estimate the amplitude of the compensatory movement at the next step, our controller generates a compensatory signal which is phase locked with the optical flow signal. It effectively tries to predict the future, as the stabilizing commands are generated at a higher frequency than the optical flow. Our system is able to track changes in the movement applied to the robot and adapt its parameters to go back to a stabilized gaze. We show that our system can be used to stabilize the gaze of a moving robot using multiple degrees of freedom in the head. Further, it can be applied to tracking objects of arbitrary shape, colors and textures moving rhythmically.

In the following sections we present the system and its properties, then explain the influence of the different open parameters of the system. We explain how to use the system to stabilize the gaze using multiple degrees of freedoms in the head, and show that it can even stabilize the gaze of a robot on a moving object. We show that the system can be applied on legged locomotion (with the Hoap2 humanoid robot walking) and non legged locomotion (with a swimming salamander robot), as long as the movement of
the head is periodic and close to a sine wave. We present
the system applied on the real Hoap3 robot tracking a
periodically moving apple.

II. PRESENTATION OF THE SYSTEM

In this section we present the details of the head stabil-
ization controller. First we present a simplified version of
the controller using a standard Hopf Adaptive Frequency
Oscillator as first developed by Buchli, Righetti and Ijspeert
([1], [9]), and then show how we adapted it to satisfy the
requirements of the head stabilization problem. Note that
the AFOs are used here in a different manner as in the
previously cited papers. Here we use the AFO in fully closed
loop (the forcing signal changes the pattern generated by the
oscillator and the oscillator’s output influences the forcing
signal, the optical flow). Furthermore, the goal here is not
to learn the shape of the forcing signal as in [10], but to
learn a correcting signal which leads to the suppression of
the forcing signal. In contrast, AFOs were previously always
used in open loop except in [1] where the AFO changes, very
slowly, the frequency of the control and thus the teaching
signal (only the frequency is changed in closed loop though,
the amplitude remains constant).

Figure 1 outlines the architecture of the system. Images
from the camera to stabilize are processed to obtain a
measure of the optical flow using the standard OpenCV
implementation of the Lucas Kanade - Shi Tomasi algorithm
([6], [12]). The optical flow signal is fed negatively to an
Adaptive Frequency Oscillator which will tune its frequency,
amplitude and phase shift so as to generate a signal phase
locked with its teaching signal (in anti phase with the optical
flow), with the correct amplitude to minimize the optical
flow. The output of the AFO is then used to control the head
of the robot. We use here a slightly modified version of the
Hopf AFO in polar coordinates in which we removed the
influence of the forcing signal on the radius of the oscillator,
to avoid divergence with high coupling terms.

The equations of the AFO are given below:

\[
\begin{align*}
\dot{r} &= \gamma(1 - r^2)r \quad (1) \\
\dot{\phi} &= \omega - \sin \phi \beta F \quad (2) \\
\dot{\omega} &= -\sin \phi \kappa F \quad (3) \\
x &= r \cos \phi \quad (4) \\
\dot{x} &= -\eta x F \quad (5) \\
\theta &= \alpha x + O \quad (6)
\end{align*}
\]

where \( r \) is the radius of the oscillator (i.e. the amplitude
of its oscillations), \( \phi \) its phase, \( \omega \) its frequency and \( \theta \) its output
here used to control the position of the head actuator. \( \alpha \)
here directly defines the amplitude of the oscillations and \( O \)
their offset. \( F \) is an external forcing signal (here the opposite
of the mean optical flow). \( \kappa \) and \( \beta \) are scaling factors for
the forcing signal. We describe the effect of these scaling
factors in Section III. Equations 1 and 2 describe a limit
cycle of radius 1. The forcing term in Equation 2 causes the
phase to synchronize with that of the forcing signal (as in a
standard forced Hopf oscillator), while a similar forcing on \( \omega \)
(Equation 3) tunes the frequency to that of the forcing signal.
When the oscillations are synchronized (same frequency and
same phase) with the forcing signal, the correlation between
\( x \) and \( F \) is maximized and \( \alpha \) starts increasing according to
Equation 5, causing the head of the robot to oscillate in
anti phase with the optical flow with increasing amplitude,
and thus decreasing the retinal slip, until the flow is about
null. All the parameters of the generated compensatory signal
are effectively learned such that they are conserved if the
forcing term \( F \) is removed. This is particularly useful to
deal with varying camera speeds, communication problems
or occlusions.

Theoretically, this system works by itself. However, the
convergence of the frequency is typically slow in the ex-
periments by Buchli and Righetti (a few hundred seconds).
This is mainly due to the fact that setting high values to
\( \beta \) and \( \kappa \) changes a lot the shape of the oscillations of the
AFO as well as issues discussed in Section III. Figure 2
shows how the output of the oscillator is modified when \( \beta \)
and \( \kappa \) are increased. When \( \beta \) and \( \kappa \) are high, the shape of
the oscillations is highly modified from the original cosine
wave. Furthermore, having too high coupling terms, when
dealing with head stabilization, would cause divergence of
the system. Indeed, since a jerky output as in Figure 2
(bottom) would cause a high optical flow which would in
term induce a higher forcing etc.

To solve this problem and obtain fast convergence of
the frequency while keeping control on the shape of the
oscillations we used two phases for the AFO. The first phase,
\( \phi_1 \), is used only to learn the frequency of the forcing signal.
The second phase \( \phi_2 \) is the actual phase of the oscillations,
and is coupled to the forcing signal for synchronization, with a different coupling term $\epsilon$. Typically we set $\epsilon \ll \kappa$ so that the shape of the oscillations is not altered too much. These two phases share the same value for $\omega$, so that the frequency learned is reflected on the oscillations of the head. Note that this system is equivalent to an AFO passing its frequency to a Hopf oscillator, and thus the proofs of convergence of AFOs in [9] remain valid and the properties of the Hopf oscillator are conserved.

The equations of the final system become:

\begin{align}
\dot{r} &= \gamma(1-r^2)r \\
\dot{\phi}_1 &= \omega - \sin \phi_1 \beta F \\
\dot{\phi}_2 &= \omega - \sin \phi_2 \epsilon F \\
\dot{\omega} &= -\sin \phi_1 \kappa F \\
x &= r \cos \phi_2 \\
\dot{\alpha} &= -\etaxF \\
\dot{\theta} &= \alpha x + O
\end{align}

### III. PARAMETER TUNING

In this section we study the influence of the parameters $\kappa$ and $\beta$ on the convergence of the frequency of the system. Note that $\epsilon$ only acts on the second phase $\phi_2$ which has no influence on the frequency modulation. Figure 3 shows the results of systematic tests monitoring the convergence time and the error after convergence for varying values of $\kappa$ and $\beta$. The forcing signal used for this experiment was obtained by recording the optical flow when rotating a simulated camera in the air in a texturized scene around its pitch axis with a frequency of 2Hz (in the Webots robotics simulator [7]), and normalizing its amplitude. We used eight instances of our oscillator initialized at eight different frequencies uniformly distributed around the desired frequency.

The convergence time $T_c(S)$ and error after convergence $E_c(S)$ of a signal $S$ (here the optical flow) to a desired value $s$ are defined as follows:

\begin{align}
T_c(S) &= \min(t) \forall t > T_c(S), |S(t) - s| < \lambda \\
E_c(S) &= \frac{1}{T_f(S) - T_c(S)} \int_{T_c(S)}^{T_f(S)} |S(t) - s| dt
\end{align}

where $T_f(S)$ is the final time of the signal $S$ and $\lambda$ is a chosen small value (in this study we used $\lambda = 0.25$). In clear, $T_c(S)$ is defined as the minimum time after which the signal $S$ stays bounded in a neighborhood of a desired value $s$, and $E_c(S)$ as the mean of the instantaneous distance between $S(t)$ and $s$ after $T_c(S)$. These two quantities are then averaged over the eight oscillators.

The error after convergence $E_c(S)$ (Figure 3, right) is basically proportional to $\kappa$, although it slightly decreases when $\beta$ is increased for a given value of $\kappa$. The convergence time $T_c(S)$ (left) decreases monotonically with $\kappa$ and meets a minimum for a specific value of $\beta$ which depends on the value of $\kappa$. This minimum is however less visible when $\kappa$ increases.

Figure 4 shows the evolution of the frequency of the system for characteristic values of $\kappa$ and $\beta$ and for different initial frequencies. For small values of $\kappa$ and $\beta$ (Figure 4a), the convergence takes a long time, especially for initial frequencies far away from the frequency of the forcing signal, while the remaining oscillations after convergence have very small amplitude. When $\beta$ and $\kappa$ are increased (Figures 4b and 4c), the convergence time decreases but the oscillations after convergence amplify. Increasing only $\beta$ (Figure 4d) has a smoothing effect on the convergence. The AFOs with initial frequency far away from that of the forcing signal converge faster, while the others converge more slowly. Increasing $\kappa$ while keeping $\beta$ low (Figure 4e) causes the convergence to be very jerky, and increases the amplitude of the remaining oscillations at convergence compared to when both parameters are set high (Figure 4c). Figure 4f shows an example of a compromise between convergence speed and error after convergence.

This study will serve as a reference to choose the values of these parameters depending on whether convergence speed or precision at convergence is more critical, but also depending on whether we can have a good estimate of the frequency of the head movement (in which case we can afford to set lower values for $\kappa$ while still converging fast enough). Note that only two parameters need to be tuned ($\beta$ and $\epsilon$ can be fixed once and for all, they do not influence the convergence speed or quality). Also note that for any value of $\kappa$ and $\beta$ tested, the system converges, so the values of these two parameters is not too critical, but only define the quality of the stabilization. Typically during locomotion and especially for statically stable gaits, the frequency of the head motion is nearly that of the controlled robot motion, so one would want to initialize the oscillator frequency with this value. In Section V however, we show that in the case of the salamander robot swimming and the Hoap2 robot walking, this is not true for the pitch axis.

### IV. EXTENSION TO MULTIPLE AXIS STABILIZATION

So far we have only considered one oscillator, for a single degree of freedom. However the system is fairly easy to extend to multiple degrees of freedom for the head. Typically one would use one AFO per degree of freedom. The only constraint here is finding the right forcing signal for each AFO.

To result in a successful head stabilization, the forcing signal for one degree of freedom should:
and $F_y$

$$F_p = -\frac{1}{K} \sum_{i=1}^{m} \sum_{j=1}^{n} F_{ij}$$  \hspace{1cm} (16)$$

$$F_y = -\frac{1}{K} \sum_{i=1}^{m} \sum_{j=1}^{n} F_{ij}$$  \hspace{1cm} (17)$$

$$F_r = -\left( \frac{1}{K_{l}} \sum_{i=1}^{m} \sum_{j=1}^{n} F_{ij} - \frac{1}{K_{r}} \sum_{i=1}^{m} \sum_{j=1}^{n} F_{ij} \right)$$  \hspace{1cm} (18)$$

where $K$ is the number of non zero flow vectors in the whole image, $K_{l}$ and $K_{r}$ are the numbers of non zero flow vectors in the left and right quarters of the image, $m$ and $n$ are the dimensions of the image, and $F_{ij}^{x}$ and $F_{ij}^{y}$ are the $x$ and $y$ components of the optical flow vector computed at position $(i, j)$.

Note that these three forcing terms do not give a direct measure of the rotation speed of the head around each axis. This is not needed by our system. The forcing terms used for each axis need however to satisfy the three conditions given earlier. In our case, this implies that the pitch axis of the head moves the image approximately along its $y$ axis, the yaw along its $x$ axis, and that the roll rotates the image around its center. In the case of a head with two cameras on each side for instance, the forcing for the roll axis $F_r$ may not work as it is. It could be adapted by taking the difference of the flow of the left part of the left camera image and the right part of the right camera image.

**V. RESULTS**

In this section we present results of the system actually applied to the head stabilization problem. All the experiments described below have been carried out using Webots [7], a physics based simulator for robotics. Here we only actuate the head of the robot (not the eyes) but applying it to the eyes also should be straightforward. The camera is a simulated pinhole camera with a field of view of 45° and providing an image of 320 x 240 pixels at 20Hz. (which is below standard for robotics cameras). The optical flow is computed from the camera images the same way in simulation and on the real robot, and is thus subject to noise (image noise, processing artifacts etc.). The reader is advised to refer to the video attached to this paper for a better insight of the following experiments.

Figure 5 shows the evolution of the frequency and the amplitude of the system when a robot (here the Fujitsu Hoap 2 humanoid robot) is rotated in the air with sine waves of different frequencies for the pitch, roll and yaw axis (see Figure 6d). One instance of our oscillator is used per degree of freedom with different forcing signals as explained in Section IV. To demonstrate the self tuning ability of the system, the frequency of the motion for the pitch axis is set arbitrarily to 2Hz, for the roll 0.75Hz and for the yaw 1Hz. At $t = 15s$, the frequencies are switched to: pitch axis: 1Hz, roll axis: 1.5Hz, yaw axis: 2Hz. The AFO is initialized with a frequency of 0.5 Hz.
The frequency of the AFOs controlling each actuator of the head quickly converges to that of the motions applied to the robot and the amplitude starts increasing until the optical flow is minimum. When the frequencies of motion are suddenly altered, the system tracks the change of frequency and recovers until the optical flow is minimal again. The resulting flow after convergence is reduced to less than 1 pixel/frame both times, in about 10 seconds.

Our system does not assume that the movement to compensate is a rotation. It actually works even for pure translations. Figure 6 describes a similar experiment as the previous one, but this time with the robot periodically translated along the $x$ and $y$ axis (the $y$ axis here is the vertical, while the $x$ axis is sideways with respect to the robot) with sine waves of different frequencies: for the $x$ axis $1\text{Hz}$, for the $y$ axis: $2\text{Hz}$. Again the system converges quickly leading to lateral and vertical head movements that almost completely suppress the optical flow. After the switch in frequency, the system recovers and goes back to nearly perfect stabilization.

As explained in Section II, our system generates oscillations whose shape can be slightly modified by the forcing signal, but remains close to a sine. Figure 7 shows the behavior of the system when the robot is rotated around its pitch axis with waves of different shapes. For every shape, the system manages to learn the main frequency of the optical flow signal. It also manages to reduce the optical flow, leading to a more stabilized gaze than without the system. However, the further the shape of the rotation is from a sine, the worst the performance, as expected.

Figure 8 shows the performance of the system when the robot is rotated around its pitch axis with a chirp ($\sin(2\pi(\omega_0 + kt)t)$), first with a relatively slow changing frequency, and then with a much faster changing one. When the frequency of the robot rotation is changing slowly, the system is able to track these changes fast enough to enable good gaze stabilization. When the frequency of the movement is changing faster, the system still tracks it but not fast enough to lead to optimal performance stabilization. Figure 8 also shows the actual range of effectiveness of the system. For both cases, at frequencies higher than $5.5\text{Hz}$ the system is not able to find the frequency of the teaching signal and the stabilization does not work anymore. However, it is important to note that this limit is not intrinsic to Adaptive Frequency Oscillators, which have an infinite basin of attraction. This limit is simply due to the low sampling rate of the optical flow, causing the signal to noise ratio to
be very high at 5.5Hz

Figure 9 shows the behavior of the system in the presence of external perturbations. The robot is successively perturbed by applying random rotations and translation for a short period of time (0.2s at t = 10s) and then for a longer period (2s at t = 15s). When the perturbation is short, the frequency and amplitude of the oscillator hardly change at all, and the stabilization recovers quickly. When the perturbation lasts longer, the frequency deviates and the amplitude drops dramatically. When the perturbation stops, the system relearns the frequency of the optical flow and restabilizes the head.

Our system relying only on visual cues, it can also be used to stabilize the gaze of the robot on periodically moving objects of arbitrary shapes, colors etc. Figure 10 shows results of the system applied to the Hoap 2 robot tracking a sphere (the moon) being translated with a sine wave along the x and y axis (vertical and sideways). The robot is not moved in this experiment. The frequencies of the motion of the sphere along the x and y axis are set respectively to 2Hz and 1Hz. At t = 15s, the frequencies are switched to 1Hz for the x axis and 2Hz for the y axis. The system is able to stabilize the gaze of the robot on the object almost perfectly, and tracks changes in the movement of the object. The result is the object staying almost exactly in the center of the image after convergence (about 5s).

We now show the system applied to robot locomotion. Figure 11 shows the evolution of the frequency and the amplitude of the oscillators controlling the pitch, roll and yaw axis of the camera attached at the tip of the head of a simulated salamander robot swimming. The salamander robot ([3]) is a modular 12DoF robot controlled with coupled oscillators (central pattern generators), and is capable to switch from walking to swimming. For this experiment, it is swimming by generating a traveling wave along its body whose frequency and amplitude can be modulated. The frequency of this wave is initially set to 1Hz. At t = 30s, the frequency is switched to 1.5Hz. The frequency of the oscillators is initialized to 0.5Hz. Again the system successfully stabilizes the gaze of the robot along the two axis, and tracks the change of frequency of the motion. The remaining optical flow after convergence is due partially to the forward motion of the robot, as shown in Figure 11d.

In the case of the salamander swimming, we could have initialized the frequency of the head stabilizing oscillator to the frequency of the motion control (we did not to demonstrate the tuning abilities of the system). Note however that the frequency of the motion of the head around the pitch axis is twice that of the general motion of the robot. The head is diving in the water at each half period of the traveling wave controlling the robot. This particularity is highly related to the gait used here and is very difficult to predict a priori (it would need complex modeling of the fluid dynamics). Our system however learns the correct frequency for this axis without the need of any modeling. Figure 12 shows snapshots of the salamander swimming, with and without the head stabilization system enabled.

Figure 13 shows a similar experiment as above with the Hoap2 walking. Three axis stabilization is used in the same
VI. CONCLUSIONS AND FUTURE WORKS

In this paper, we presented a novel approach to the head stabilization problem during periodic movements. Our system uses only visual cues, here optical flow, to stabilize the head of a robot subject to periodic motion, typically during locomotion. The system tries to predict the motion of the robot, by learning the frequency, phase and amplitude

way as for the other experiments. The robot is controlled using the default gait provided by Fujitsu. The frequency of the motion is not altered for this experiment, since the gait is only stable with the precomputed parameters. The frequency of the AFOs is initialized to 1Hz and at \( t = 30s \) the frequency is switched to 1.5Hz. The AFO is initialized with a frequency of 0.5Hz. Figure 11d shows the evolution of the norm of the mean optical flow vector over time as well as the flow due to the forward motion of the robot.

Fig. 11: Evolution of the frequency and the amplitude (\( \alpha \)) of the oscillator when the salamander robot swimming. The frequency of swimming is initially 1Hz and at \( t = 30s \) the frequency is switched to 1.5Hz. The AFO is initialized with a frequency of 0.5Hz. Figure 11d shows the evolution of the norm of the mean optical flow vector over time as well as the flow due to the forward motion of the robot.

Optical flow in pixel / frame

(a) pitch axis
(b) roll axis
(c) yaw axis
(d) optical flow

Fig. 12: Snapshots of the salamander robot swimming without gaze stabilization (a, b, and c) and with gaze stabilization (d, e and f). When the camera stabilization system is enabled, the gaze (highlighted by the purple camera frustums) always points in the direction of motion.

We performed the object tracking experiment with the real Hoap3 robot, which has embedded cameras in its head. An apple was attached to a spring, allowing it to swing horizontally and vertically, with different frequencies. We used here the exact same system as in the experiments in simulation. Taking into account the framerate of the camera, the computation time of the optical flow and communication delays, we can provide our oscillator with visual forcing at a frequency of about 10Hz. Figure 14 shows the evolution of the frequency and amplitude of the two axis controlling the head. Here, the optical flow was not a good measure of the performance of the system, due to the high noise even after stabilization (see attached video). Instead we used simple blob tracking to compute the position of the apple in the image frame (Figure 14c). Even with such a slow and noisy optical flow, the system is able to stabilize the object around the center of the image. Around \( t = 45s \), the stabilization around the yaw axis gets worse for a couple of seconds, but quickly recovers. Note that the frequency of the apple motion is not perfectly constant here due to the natural damping of the spring and the air friction.

Fig. 13: Evolution of the frequency and the amplitude (\( \alpha \)) of the oscillator when the Hoap2 robot is walking. The AFO is initialized with a frequency of 2Hz. Figure 13d shows the evolution of the norm of the mean optical flow vector over time. Figure 13e shows the shape of the rotation speed of the base of the head.

By the way, we have the real Hoap3 robot, which has embedded cameras in its head. An apple was attached to a spring, allowing it to swing horizontally and vertically, with different frequencies. We used here the exact same system as in the experiments in simulation. Taking into account the framerate of the camera, the computation time of the optical flow and communication delays, we can provide our oscillator with visual forcing at a frequency of about 10Hz. Figure 14 shows the evolution of the frequency and amplitude of the two axis controlling the head. Here, the optical flow was not a good measure of the performance of the system, due to the high noise even after stabilization (see attached video). Instead we used simple blob tracking to compute the position of the apple in the image frame (Figure 14c). Even with such a slow and noisy optical flow, the system is able to stabilize the object around the center of the image. Around \( t = 45s \), the stabilization around the yaw axis gets worse for a couple of seconds, but quickly recovers. Note that the frequency of the apple motion is not perfectly constant here due to the natural damping of the spring and the air friction.
of the optical flow. All the learning is done online, and embedded into the dynamics of the designed oscillator such that changes in the parameters of the motion are tracked by the system. We showed that our system can be applied to stabilize the gaze when the robot is being moved, or when it is tracking a periodically moving object. We also showed that the system can successfully stabilize the head of the robot during actual locomotion, with a biped and an anguilliform robot, without the need for fast sensors. We demonstrated the performance of the system on a real robotics setup. More experiments on real robot should follow. Another planned experiment would consist in stabilizing an actuated camera attached to a real human or animal during locomotion.

The main limitation of the current system is the single shape of the output of the oscillator. We assume that the necessary compensatory motion to stabilize the head is close to a sine wave (note that the optical flow does not need to be sine wave, as long as the compensatory movements are). This is not always the case during locomotion. The shape of the oscillations is slightly modulated by the feedback term, but being able to generate the exact right oscillation patterns would surely increase the performance of our system. Future work should include learning the whole shape of the robot motion. This could be done for instance by deducing the shape of the compensatory signals from the optical flow, and adding a dynamical filter to our oscillator (for instance of a combination of sine waves with different frequencies and amplitudes), or by designing an adaptive Gaussian mixture filter. Using a pool of coupled Adaptive Frequency Oscillators as done in [10] would be another solution to generate more complex shapes for the head motion.

The approach described in this paper uses only vision as sensory feedback. This is a big advantage of the system since it can be applied to a wide variety of robots which do not necessarily have a large set of sensors. However, this is not a limitation of the system, and one could imagine using information from more sensors, depending on the application. A simple way to fuse information from a vestibular system and cameras, for instance, could be to use two different forcing signals for our oscillator. The forcing in Equation 8 would come from the vestibular system while the one in Equation 9 would come from visual cues. This would increase the speed and smoothness of the convergence of the frequency, while keeping the head motion phase locked with the optical flow.

Finally, let us note that only the parameters of the nominal gait of the robot are learned. Unexpected fast changes of the head motion pattern are not stabilized. For some applications, this could be a downside of the system. However, during locomotion, fast changes of optical flow after head stabilization could be a sign that some unexpected events are occurring, e.g. the robot loosing balance, and this information could be used to trigger a response in a higher level controller.

VII. ACKNOWLEDGMENTS

This work was funded by the IST-EPFL Joint initiative and partially by the EU Commission within the Seventh Framework Programme FP7, under grant agreement 248366 (RoboSoM project) and 248311 (AMARSi project).

REFERENCES