Model-Based and Model-Free Approaches for Postural Control of a Compliant Humanoid Robot using Optical Flow

Sébastien Gay[1,2], Jesse van den Kieboom[1], José Santos-Victor[2] and Auke Jan Ijspeert[1]

***Abstract*— Vision is a very rich sensor with a proven critical role in the control of balance. However, it is widely under-used for robotics postural control. This paper presents and compares two approaches, one model-based and one model-free, to ensure stability of the COMAN compliant humanoid robot standing on a moving platform. The model-based approach uses inverse kinematics, while the model-free one relies on a neural network as mapping between sensors and actuators. The sensory information is composed of proprioceptive cues (gyroscope) and visual cues, used separately or together. We present methods of using vision as sensory input without relying on a particular object or feature of the scene, but only on the optical flow. The performance of both approaches are compared systematically in a realistic robotics simulator, for different movements of the platform and using different sensory cues. We aim to see if vision can replace proprioceptive sensors or be fused with them to improve the performance of the stabilizing controller. While both model-based and model-free approaches successfully stabilize the robot, the model-free approach shows better overall performance. Preliminary results on the real COMAN robot are shown.**

I. INTRODUCTION

Vision is probably the richest sensor available to most animals and humans. It provides of course extroceptive information about the surrounding environment, but also ex-proprioceptive cues about ones own kinematics and position in the world. In the last decades, vision, mostly optical flow, has been shown to play a role in unsuspected aspects of human motion like finely tuning the trajectory of the foot when stepping over an obstacle ([15]), estimating the travelled distance ([11]) and modulating walking speed and gait transition ([10]). For balance control, vision has even been shown to override the other sensory cues ([16], [8]). More specifically, optical flow has been identified as the main actor in the control of posture ([2]). In robotics however, vision is generally highly under-used for the control of balance, other sensors as inertial measurement units (IMU) or force sensors being preferred. Most modern approaches for balance control rely on the Zero Momentum Point (ZMP) ([18]), or the Center of Pressure, which are actually the same point viewed from two different perspectives ([6]). The ZMP is typically used to ensure stability maintaining the center of gravity inside the support polygon. Alternatively, stability can be achieved by relying on the kinematic model. In [7] the authors designed kinematic sinergies for the lower part of the Hoap2 robot which linearize the balancing control problem. Another kinematic based approach is Virtual Model Control ([17]) which attaches virtual springs at different points of the robot and projects their generated forces through the Jacobian of the robot to the different joints. In [13], an optimization based method inspired form grasping is applied to balancing the DLR-KUKA humanoid robot. By distributing a force and torque among previously selected contact points on the feet, the robot is able to maintain a desired center of mass position.

[1]Biorobotics Laboratory, Ecole Polytechnique Fédérale de Lausanne
[2]VisLab, Institute for systems and robotics, Instituto Superior Técnico, Lisbon

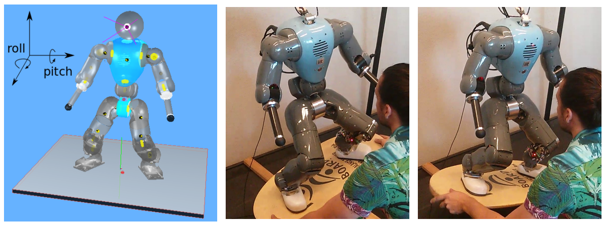

Fig. 1: The problem tackled in this paper: the compliant humanoid COMAN is standing on a moving platform and should maintain stable posture using either the gyroscope, the optical flow, or both.

To the best of our knowledge, approaches using vision for balance control are seldom. In [12] and [14], visual information is used to estimate the position of the ZMP and achieve robot stability. However both approaches rely on a reference object to estimate the pose of the robot.

In this paper we present two approaches to the balance control problem of a compliant humanoid robot, the CO-MAN, standing on a moving platform. Figure 1 illustrates the problem we consider here.

The first approach presented here is a model-based kinematic approach using closed-form equations for each leg and ensuring that the feet do not slip while the trunk remains upright. The second one is a model-free optimization approach based on an artificial neural network mapping sensor values to joint velocities and integrators for the low level control. In addition to the visual information, we also consider gyroscope values as sensory input in this paper. Unlike the work mentioned before, the visual information here is not based on a particular object of the scene or special features like the horizon orientation, but relies solely on the optical flow, which is very general and can be computed in nearly any environment. The goal of this paper is also to compare the model-based and model-free approaches and investigate if vision can replace or be fused with a gyroscope to increase performance.

In Section II and III, we present our control approach for model-based and model-free control of balance. Then, Section IV explains how the camera images are processed to extract relevant information for each of the two approaches. Finally Section V presents experiments performed in Simu-

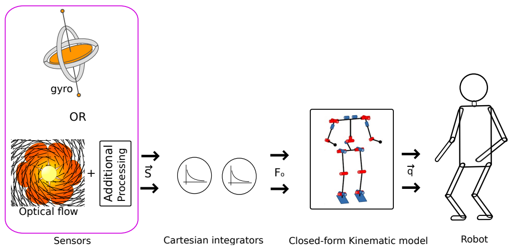

Fig. 2: The general control framework of the model-based approach. The absolute orientation of the platform is estimated from either the gyroscope values or the camera images. A closed form kinematic model controls the position of each joint so as to adapt the feet orientations and the leg lengths to keep the trunk upright and ensure that the feet do not slide on the platform.

lation and on the real COMAN robot.

II. Model-based control framework

The general control framework for the model-based postural control of the COMAN robot is depicted in Figure 2. COMAN is a 94.5 cm tall 25 DoF humanoid robot weighting 31.2kg. It features series elastic actuators in the shoulders, hips, knees and ankles.

The pitching and rolling rotation speeds of the trunk of the robot are estimated using either the embedded gyroscope of the robot, or a camera placed on its neck. The process of computing the rotation angles using a camera is explained in Section IV. The rotation speeds are then filtered and integrated to extract an estimation of the absolute orientation of the platform (a simple Euler integration is used). This implicates that the initial posture of the robot has to be upright. We call α_p and α_r the absolute pitch and roll angles of the platform and $\hat{\alpha}_p$ and $\hat{\alpha}_r$ their estimated value.

The absolute pitch and roll angles of the platform α_p and α_r fully determine the transformation matrices of the feet in the base reference frame of the robot such that the trunk is vertical. The transformation matrix T of the left and right foot is given below:

$$T = \begin{bmatrix} C_rC_y & -S_r & C_rS_y & p_x \\ S_pS_y + C_pC_yS_r & C_pC_r & C_pS_rS_y - C_yS_p & p_y \\ C_yS_pS_r - C_pS_y & C_rS_p & C_pC_y + S_pS_rS_y & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where $C_i = cos(\beta_i)$, $S_i = sin(\beta_i)$ and β_p, β_r and β_y are the pitch, roll and yaw angles of the feet being defined as $\beta_p = \hat{\alpha}_p$, $\beta_r = \hat{\alpha}_r$ and $\beta_y = \pm 25°$. This value of $25°$ is arbitrarily chosen to achieve a natural looking and stable posture, and its sign depends on the left or right foot. p_x, p_y and p_z are coordinates of the foot position in the root reference frame of the robot. The leg lengths should also be adapted to compensate for rolling motion of the platform, by folding the knees and adapting the other joint angles accordingly. These are computed so as to ensure that 1) the trunk keeps its initial orientation (upright, slightly bent forward) and 2) the feet do not slide on the ground. Geometrically we can deduce the target position $\mathbf{p} = [p_x, p_y, p_z]^T$ for the left and right feet according to the initial posture and the estimated platform rotation:

$$\mathbf{p} = \begin{bmatrix} p_x^0 \cos\hat{\alpha}_r \pm L_{L1}(\cos\hat{\alpha}_r - 1) \\ p_y^0 + (p_x^0 \pm L_{L1})\sin\hat{\alpha}_r \\ p_z^0 \end{bmatrix} \quad (2)$$

where $\mathbf{p^0} = [p_x^0, p_y^0, p_z^0]^T$ is the initial position of the foot in the hip root reference frame when the platform is not rotated and L_{L1} is half the waist width of the robot. The signs are chosen accordingly for the left and right foot.

Once the target positions of the feet are chosen, various inverse kinematics methods can be used to achieve them. The most common ones rely on Jacobians, or use non-linear numerical solvers. This however introduces delays in the computation and can be computationally expensive. However since each leg is a 6-DoF manipulator and we now fixed the position and orientation of the end effector (the foot), closed form equations can be derived. Various methods exist to derive these equations usually involving computing the symbolic inverse transformation matrix, and squaring, adding and dividing some of its components to decouple the degrees of freedoms as much as possible. This can be a difficult and time consuming process and papers have been published on the sole matter of deriving closed form equations for the inverse kinematics of humanoid robots ([3], [1]). Furthermore, the order of the joints of the COMAN robot (hip pitch, hip roll, hip yaw, knee, ankle roll, ankle pitch) is not standard for humanoid robots and makes it difficult to apply these methods. Fortunately, a library named IKFast, part of the open source framework OpenRave, has been developed which is capable of computing the closed form equations for any 6-DoF manipulator and for special cases up to 8-DoF. It takes care of all the singularities and, due to redundancies, outputs up to six solutions for the inverse kinematics. These solutions are then narrowed down to one by taking into account the joint limits.

III. Model-free control framework

The general control framework for the model-free postural control of the COMAN robot is depicted in Figure 3. This framework is a derivation of the system previously presented in [5], where it was used to control a quadruped walking on rough terrain. It is worth noting that the same system can be used on different robots with a different number of degrees of freedom and for both rhythmic and discrete tasks, with only very minor modifications. In that case the low level control was performed by a central pattern generator, and all joints were coupled except the abduction/adduction. The system presented in this paper is the discrete movement version of the one presented in [5], equivalent to having the amplitudes and phase of each oscillator set to 0 and using only their offset. Thus the components controlling the joint positions here are called *integrators* instead of *oscillators* since they do not effectively oscillate but output discrete trajectories in a smooth way. Their equations can be reduced to :

$$\dot{o} = \kappa_o \mathcal{F}_o(\mathbf{S}) \quad (3)$$

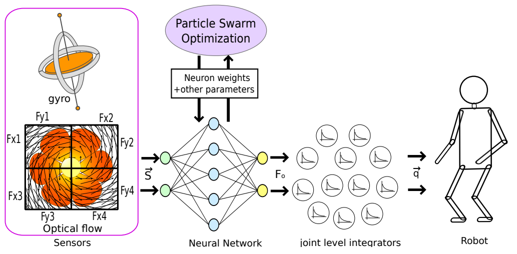

Fig. 3: The general control framework of the model-free approach. The sensory information composed from down-sampled optical flow and/or gyroscope values is fed to a neural network outputting feedback to integrators for each joint. The weights of the neural network are optimized using particle swarm optimization.

Here o directly determines the output (the joint position of the oscillator).

The sensory information consists of the gyroscope values and a down-sampling of the optical flow computed as described in Section IV. As in the model-based approach (Section IV), we could use the estimates of the rotational velocities as the input to the neural network. However, the neural network can deal with less explicit sensory input like the raw optical flow, which would save one sensor processing step. Furthermore the neural network may also extract other relevant information from the flow field. By the nature of our system, gyro and camera can be used separately or fused seamlessly. Since only rotational velocity information is used, as for the model-based approach, the controller has to be started with the robot upright.

These sensor values are used as input to a fully connected neural network with sigmoid activation, which outputs the values $\mathcal{F}_o(\mathbf{S})$ for each joint. Hence this neural network can be viewed as a learnable non-linear mapping between sensor values and joint velocities. Here, we chose to control 6 DoF for each leg and 2 DoF (shoulder hip and roll) for each arm, so 16 DoF in total. The sensor vector consists of 3 values for the gyroscope and typically 8 values for the down sampled optical flow (x and y components for each quarter of the image). The integrators for the joints take these functions as sensory feedback and output joint positions for each joint. The weights of the neural network W^D are tuned by an unsupervised learning process, using Particle Swarm Optimization (PSO). Other non-convex optimization algorithms like Genetic Algorithms or Simulated Annealing could of course also be used but we had good experience with PSO in rough fitness landscapes, as considered here. Note that we cannot use supervised learning methods such as back-propagation here because we do not know the function to be learned (the different joint trajectories to achieve stabilization). We only know that the robot should maintain upright posture and not fall, but not how. Together with the weights of the neural network, the convergence rate of the integrators and the slope of the sigmoid of the neurons are tuned since they also determine the influence of the feedback on the output trajectories. The total parameter vector for the optimization is: $[W^D, \kappa_o, \lambda]$, λ being the slope of the neurons sigmoid activation function $\frac{1}{1+e^{-\lambda x}}$

Finally, we define the fitness function used by PSO. Here

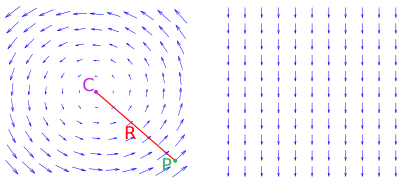

Fig. 4: A representation of the components of the optical flow caused by rolling (left) and pitching (right) rotations.

we want to maximize the time before falling. To give the optimization process some measure of the stability of the robot, we also include in the fitness function the averaged absolute pitch and roll angles to minimize. The fitness function used here is:

$$\mathcal{F} = \tau \left(\frac{1}{1 + \frac{1}{\tau} \int_{t=0}^{\tau} |\theta_P(t)| dt} \right)^{\xi} \left(\frac{1}{1 + \frac{1}{\tau} \int_{t=0}^{\tau} |\theta_R(t)| dt} \right)^{\xi} \tag{4}$$

where $\theta_P(t)$ and $\theta_R(t)$ are the absolute pitch and roll angle of the robot at time t, and τ is the time elapsed before the robot falls. ξ is a constant used to give more or less importance to the minimization of the angles with respect to the time to fall τ.

One of the advantages of using a direct mapping from sensor values providing speed cues (gyroscope and optical flow) and joint velocities is that the neural network should in principle be able to model an estimate of the robot dynamics while the model-based approach solely describes its kinematics. As a consequence, the model-free approach should cope with the compliance and inertia of the robot better than the kinematic approach.

IV. VISUAL-PROCESSING

Images from the camera are processed so as to obtain optical flow information. We use the Lucas-Kanade ([9]) implementation in OpenCV to compute the optical flow. This is a very high dimensional information (typically around 500 vectors here), and needs to be reduced. For the model-free approach, we want to investigate if the neural network can process non-explicit information like the angular rates of the gyro or the raw optical flow. Thus we just need to down-sample the optical flow to tractable values in term of number of neural network input. Here we choose to split the image into two by two quarters and compute the averaged flow in each of them. This leads to four vectors having two components each, so eight values.

For the model-based approach however, this is not enough since we need to have an accurate estimation of the absolute platform rotation. Here the optical flow can be decomposed into the part produced by the pitching motion and the one caused by the rolling motion of the camera. Since the robot is standing on a platform only subject to rotations, the translational part of the optical flow is neglected here. Figure 4 shows what these components look like.

As shown in [4], each vector of the optical flow V satisfies the following equations:

$$V = V_r + V_p = \begin{bmatrix} 0 \\ k\omega_p \end{bmatrix} + \begin{bmatrix} -\omega_r(y_p - y_c) \\ \omega_r(x_p - x_c) \end{bmatrix} \tag{5}$$

where ω_p and ω_r are the pitching and rolling rotation speeds, and k is a constant depending of the characteristics of the camera. $\mathbf{p} = [x_p, y_p]^T$ is the origin of the considered vector in the image coordinates, and $\mathbf{c} = [x_c, y_c]^T$ the center of the rolling rotation.

Subtracting vector flows pairwise we can decouple the rolling component of the flow, since the pitching component is constant over the whole image.

$$\Delta V = \begin{bmatrix} -\omega_r \Delta y \\ \omega_r \Delta x \end{bmatrix} \tag{6}$$

Each pair of vector satisfying this equality, we can apply a simple mean square error regression to estimate ω_r, and successively get ω_p.

V. Results

In this section we present results using the model-based and the model-free approach in simulation. Preliminary results using the real robot are given in Section V-C. We use the Webots simulation environment with a very realistic model of the COMAN robot. This model simulates the series elastic actuators and whole body measurements on the robot and the simulator displayed very similar behavior. The real COMAN robot does not have a camera or even a head to this date. We therefore added a massless head and camera to the model. To match those achievable on the real robot, the time step of the controller and the gyro data was set to 8ms (125Hz), and that of the camera to 24ms (~40Hz).

For both approaches the testing environment was the same. We placed the robot on a platform rotated according to the following rules:

$$\alpha_p = A \sin(2\pi F_p t) \tag{7}$$
$$\alpha_r = A \sin(2\pi F_r t) \tag{8}$$
$$A = \eta t \tag{9}$$

where F_p and F_r are the pitching and rolling frequency, and A the amplitude of the oscillations. The amplitude gradually increases with time so that the stabilization problem becomes harder. Here we set $\eta = 0.007$ so that at $t = 20s$ $A \approx 8°$ (total amplitude of 16 degrees).

A. Model-based approach

We ran simulations with every combinations of pitching and rolling frequencies ranging from 0.2Hz to 1.1Hz with steps of 0.1Hz (100 tests in total). For each simulation we recorded the trunk pitch and roll angles and the time to fall. The results are presented in Figure 5. Small angles mean that the trunk is upright, so that the stabilization is more successful. Large time to fall values mean that the controller is able to cope with larger movements of the platform, since their amplitude is gradually increasing with time. Since the total simulation time is 20s, a time to fall of 20 is considered a success (the robot did not fall). When not controlling the robot, the success rate is 14%. This corresponds to low frequencies, where the initial posture is stable enough to cope with the movements of the platform. The success rate of the

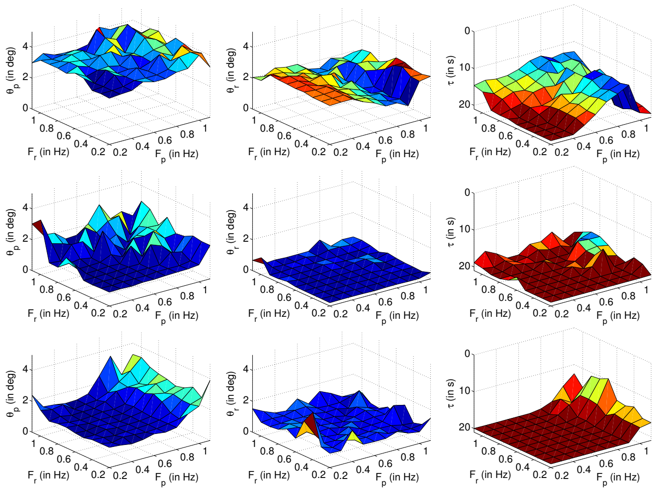

Fig. 5: Average trunk pitch (left column), average trunk roll (middle column) and time to fall (right column, z axis inverted) without control (top row), with the model-based controller using gyro values (center row) and with the model-based approach using camera (bottom row). All tests lasted 20 seconds.

model-based controller using gyro as input over all the trials was 60%, while the one using vision as input reached 71%. However, the controller using gyro input was more successful than the one using optical flow at reducing the rolling and pitching motion with an averaged pitch angle of 1.47 degrees against 1.69 degrees and an averaged rolling motion of 0.27 degrees against 0.98 degrees.

B. Model-free approach

The model-free approach requires a phase of training. This phase consisted in running a PSO algorithm with 100 particles for 300 iterations maximum. Each particle is a set of parameters (weights, sigmoid slope and convergence rate) defining the neural controller. We tried different training scenarios. For the first scenario, we trained the controller with a platform pitching frequency of 0.7Hz and a rolling frequency of 1Hz. For the second one, the controller was trained simultaneously with three combinations of pitching and rolling frequencies : 0.3Hz - 0.7Hz, 0.7Hz - 0.5Hz and 1Hz - 1Hz. This means each PSO particle was repeated three times and its fitness was computed as the average of the fitness for each frequency combinations. The last scenario consisted in training the controller with six combinations of frequencies: 0.3Hz - 0.7Hz, 0.7Hz - 0.5Hz, 1Hz - 1Hz, 1Hz - 0.3Hz, 0.5Hz - 0.3Hz and 0.7Hz - 1Hz.

Each scenario was repeated using either only the gyro as sensory input, or the camera, or camera and gyro together. To make sure results do not depend on initial conditions of the PSO algorithm, we repeated each scenario three times, and got qualitatively similar results.

The evolution of the maximum fitness function for each scenario using the gyro as sensory feedback is shown in Figure 6. The optimizations take between 150 and 300 iterations to converge but the more repetitions the training has, the longer the optimization takes to converge. The same property can be observed with camera or camera and gyro as sensory input.

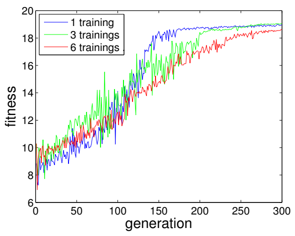

Fig. 6: Evolution of the fitness for each of the 3 scenarios: training once, training 3 times and training 6 times

Finally we ran generalization tests for each of the scenarios. As for the model-based approach, we used frequencies ranging from 0.2Hz to 1.1Hz with steps of 0.1Hz, for roll and pitch (100 tests in total). The results are shown in Figure 7. We define success as $\tau = 20s$, meaning the robot did not fall during the whole simulation. When not controlling the robot, the success rate is 14%, as mentioned in Section V-A. When the robot is controlled with the neural network trained with only one pair of frequencies, the performance climbs to 73%. When increasing the number training frequencies to three the success rate actually decreases to 61%. This decrease of performance between the first and second scenario is surprising, since in principle increasing the number of training sample should increase the generalization score (for a very high number of training samples, over-fitting problems might arise but this is clearly not the case here with only three pairs of frequencies). Actually, the generalization performance does not decrease, as defined by the fitness function presented in Section III. Indeed, while the averaged time to fall is indeed lower for the second scenario (19.02 vs 19.34 for the first one), the averaged rolling and pitching angles are also lower (0.56 vs 0.64 degrees for the pitch and 0.25 vs 0.38 degrees for the roll), making the overall fitness of the second scenario higher. When using six pairs of frequencies for training, the performance climbs again to 90%. The controller is able to generalize to most unseen cases, even slightly outside the training bounds (frequencies of 0.2 and 1.1Hz). Furthermore, except where both the pitching and the rolling frequency were above 0.9Hz, nearly all (97%) of the trials were successful.

Figure 8 shows the same performance indicators as before for the model-free method using different sensory inputs : gyro alone, vision alone and gyro and vision together. The controller was trained with three different frequencies. Here as for the model-free approach the performance using the camera was slightly higher than the one using the gyro (64% against 61%). This is interesting considering the visual input is simply the down-sampled raw optical flow. When combining gyro and camera together, the success rate raised to 75%. The minimization of the angles was not as good when using the camera as when using the gyro only. One explanation for this might be the greater latency of the camera sensor.

C. Real robot experiments

We started implementing the model-based approach on the real robot. For now we only considered the rolling angle, since it is less subject to spring excitation by the control. We recorded the rolling angles over a 25s experiment with and without the controller. Results are shown in Figure 9. Please note that the motion of the platform was done by hand, and although we tried to apply the same amplitude of motion with and without the control, these motions obviously differ. Furthermore we had to make the frequency of the motion without control very low in order to prevent the robot from falling. The amplitude of the rolling motion with the control was reduced by more the 50% compared to not controlling the robot. The video attached with this paper shows the model-based approach applied on the real robot, and snapshot of it can be seen in Figure 1.

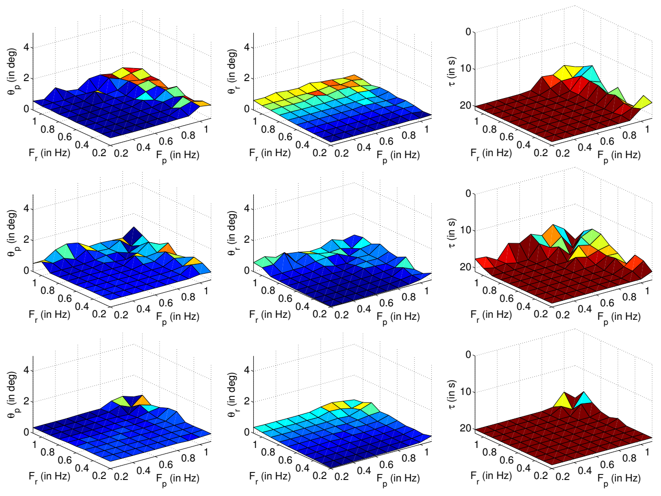

Fig. 7: Average trunk pitch (left column), average trunk roll (middle column) and time to fall (right column, z axis inverted) for each of the three scenarios: training once (top row), training with three different frequencies (center row) and training with six different frequencies (bottom row). All generalization tests lasted 20 seconds.

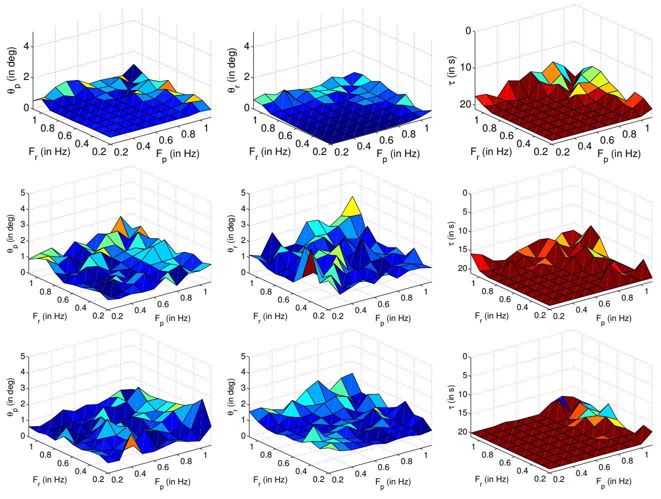

Fig. 8: Average trunk pitch (left column), average trunk roll (middle column) and time to fall (right column, z axis inverted) for the model-free approach using gyro only (top row), optical flow only (center row) and gyro with optical flow (bottom row).

VI. CONCLUSIONS

In this paper we presented two approaches to use vision, instead or in addition to a gyroscope, to control the posture of a compliant humanoid standing on a platform. The first

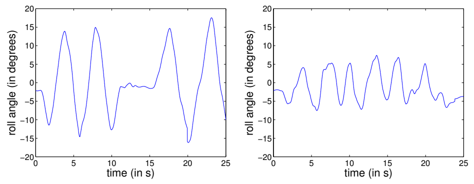

Fig. 9: Rolling angle of the real COMAN robot for a 25s experiment without control (left) and with the model-based approach (right). The motion of the platform is applied by hand, so both motions differ, but the amplitudes of the platform movement should be roughly the same.

approach is kinematic based, and ensures that the trunk stays straight up while the feet do not slide on the platform. The second one is a model-free controller based on a neural network fusing sensory information and mapping it to joint velocities. Joint integrators then output joint positions to the robot. We systematically tested these two approaches for different frequencies of the pitching and rolling motions of the platform. While the model-based approach was successful at reducing especially the rolling angle of the robot, it showed limitations when reaching higher frequencies. In contrast the model-free approach, when properly trained, improved both the time to fall and the rolling and pitching angles by a great amount. This proves that a linear kinematic control is not sufficient for controlling the posture of a compliant robot. The superiority of the neural-network approach over the model-based one can be explained by the fact that it maps sensor values representing rotational velocities of the robot, in contrast to the model-based control which relies on absolute rotation angles. Thus the neural network can in fact model some aspects of the dynamics of the robot and deal better with its compliance and inertia.

Both model-based and model-free approaches were slightly more successful when using vision than when using the gyro. One reason for that could be that the visual estimation of the rotations of the robot is less sensible to noise since it integrates a great number of optical flow vectors, while the gyro is only two values. Furthermore, the fact that stabilization is possible using the model-free approach with raw optical flow as input and no rotation estimation is an interesting feature. Fusing gyro and optical flow together significantly improved the success rate compared to using only the individual sensors.

Note that our work does not mean to imply all that model-free methods are better than kinematic based methods. We presented here one model-free method versus a relatively classical kinematic based one, and thus can only draw conclusions on these two. We believe both approaches have interesting features which make them better suited to different situations. When an kinematic model of the robot is available, implementing the model-based approach might be the best way to get stabilization working fast. When dealing with inertia and compliance of the robot is critical, the model free approach may be better suited.

For future experiments, it would be interesting to check if taking the best individual of the optimization really leads to the best generalization performance. Indeed, it is well known in the optimization community that the very best individual is often over-specialized to its training data. Instead, taking the best individual of previous iterations might show better performance in unseen circumstances. It would be interesting to investigate fusing more sensors with the neural network, like force sensors, and to compare our model-free approach to an inverse dynamics or ZMP based approach.

ACKNOWLEDGMENT

This work was funded by the Portuguese Fundation for Science and Technology (FCT) through the IST-EPFL joint initiative, EPFL's Biorobotics Laboratory, and the European Commission grant AMARSI FP7-ICT-248311.

REFERENCES

[1] M a Ali, H a Park, and C S G Lee. Closed-form inverse kinematic joint solution for humanoid robots. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 704–709, October 2010.

[2] B G Bardy, W H Warren, and B a Kay. The role of central and peripheral vision in postural control during walking. *Perception & psychophysics*, 61(7):1356–68, October 1999.

[3] S. Bertrand, O. Bruneau, F.B. Ouezdou, and S. Alfayad. Closed-form solutions of inverse kinematic models for the control of a biped robot with 8 active degrees of freedom per leg. *Mechanism and Machine Theory*, 49:117–140, March 2012.

[4] JY Chang, WF Hu, MH Cheng, and BS. Digital Image Traslational and Rotational Motion Stabilization Using Optical Flow Technique. In *IEEE Transactions on Consumer Electronics*, volume 48, 2002.

[5] S. Gay, J. Santos-Victor, and A. Ijspeert. Learning Robot Gait Stability using Neural Networks as Sensory Feedback Function for Central Pattern Generators. *Submitted to the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[6] Ambarish Goswami. Postural stability of biped robots and the foot-rotation indicator (FRI) point. *The International Journal of Robotics Research*, 1999.

[7] Helmut Hauser, Gerhard Neumann, Auke J Ijspeert, and Wolfgang Maass. Biologically inspired kinematic synergies enable linear balance control of a humanoid robot. *Biological cybernetics*, 104(4-5):235–49, May 2011.

[8] J. R. Lishman and D. N. Lee. The autonomy of visual kinaesthesis. *Perception*, 2(3):287–294, 1973.

[9] BD Lucas and T Kanade. An iterative image registration technique with an application to stereo vision. In *International joint conference on artificial intelligence*, number x, pages 674–679, 1981.

[10] Betty J Mohler, William B Thompson, Sarah H Creem-regehr Herbert, L Pick Jr, and William H Warren Jr. Visual flow influences gait transition speed and preferred walking speed. *Experimental Brain Research*, pages 221–228, 2007.

[11] Matteo Mossio, Manuel Vidal, and Alain Berthoz. Traveled distances: new insights into the role of optic flow. *Vision research*, 48(2):289–303, January 2008.

[12] Naoki Oda, Junichi Yoneda, and Takahiro Abe. Visual feedback control of ZMP for biped walking robot. *IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society*, pages 4543–4548, November 2011.

[13] Christian Ott, Maximo a. Roa, and Gerd Hirzinger. Posture and balance control for biped robots based on contact force optimization. *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 26–33, October 2011.

[14] Sangbum Park, Youngjoon Han, and Hernsoo Hahn. Balance control of a biped robot using camera image of reference object. *International Journal of Control, Automation and Systems*, 7(1):75–84, March 2009.

[15] AE Patla. How is human gait controlled by vision. *Ecological Psychology*, (786945360), 1998.

[16] Aftab E. Patla. Understanding the roles of vision in the control of human locomotion. *Gait & Posture*, 5(1):54–69, February 1997.

[17] J Pratt, CM Chew, and Ann Torres. Virtual model control: An intuitive approach for bipedal locomotion. *International Journal of Robotics Research*, (129), 2001.

[18] M Vukobratović and J Stepanenko. On the stability of anthropomorphic systems. *Mathematical Biosciences*, 37:1–37, 1972.