Unsupervised learning of camera exposure control using randomly connected neural networks

Oswald Berthold∗ Verena V. Hafner∗

∗Humboldt-Universität zu Berlin, Dept. of Computer Science, Cognitive Robotics Group, Germany (e-mail: {oberthol|hafner}@informatik.hu-berlin.de).

Abstract: We use webcams on single board computers for vision-based control of flying robots. In that context we consider autonomous acquisition (bootstrapping) of exposure and gain control policies for the digital cameras. The policies are generated by neural networks with random connectivity which can be regarded as nonlinear expansion kernels acting on the input. We consider both feed-forward and recursive networks and apply these structures to learning the required policies. The camera represents an embodied robotic subsystem which is subject to temporal delays in its response. The performance measure is based on selective regions of interest in the image. The contribution of this paper is a complete embodied autonomous learning loop.

Keywords: Robotics; Learning; Unsupervised; Neural Networks; Vision; MAV

1. INTRODUCTION

Using a digital camera on embedded Single Board Computers (SBC) is a viable approach for realizing robotic vision on a flying robot as shown in Fig. 1(a). In order for vision algorithms to work, the distribution of the pixel values, as they reflect an outer contrast pattern, needs to be matched to the system’s sensitive operating range. The camera is a transducer, converting contrast patterns mediated by incident light into analog voltages and then into a list of digitally encoded numbers. There are three basic controls in the analog domain which act together on a common resulting variable, overall image brightness. These controls are the aperture, shutter- or integration time respectively and analog amplification, which are schematically displayed in Figure 2. To avoid information loss during analog to digital (AD) conversion, the controls need to be set appropriately.

Two conditions make the camera’s built-in Auto Exposure (AE) and Automatic Gain Control (AGC) mechanism unusable for our purposes. First, we would like to know the values of the control parameters at any time so they can be used as an effenent copy within a larger control system. Second, the Regions Of Interest (ROI) in the image, to which adjustment is referenced might be comprised of arbitrary subsets of sensor pixels, as is the case when using omnidirectional cameras as in Fig. 1(b) or cropping. This issue is treated comprehensively in Nourani-Vatani and Roberts (2007).

We regard the camera control or constant image brightness problem as a simple and yet interesting instance of embodied robotic learning, as it is exemplary in terms of the closed sensorimotor loop by providing response latency and real-world noise. The complete system is pictured in Fig. 1(a).

Fig. 1. Flying robot and omnidirectional camera view. Fig. 1(a) shows the complete system comprising of quadrotor airframe, onboard computing, sensors and omnidirectional lens mounted on top. Fig. 1(b) shows how the lens projects onto the rectangular image sensor. More than half of all pixels can be disregarded when calculating camera exposure settings because they are always black.

1.1 Related work

AE and AGC are integral parts in the inventory of consumer and professional photography and video capture devices. Mostly, these solutions build on large datasets from which lookup tables for control values are extracted. Recently, techniques such as multi-slope cameras, bracketing and High-Dynamic-Range (HDR) photography brought additional challenges to exposure and focus control. These include the need for localized reference in the image, according to where ROIs, for example human faces, are detected to realize optimal exposure in those particular areas.

Several solutions for robotic exposure control based on conventional techniques have been proposed. In Nourani-
2. METHODS

2.1 Bootstrapping methodology and problem statement

Given a camera with known control parameters but unknown transfer characteristics, we want to find a mapping \( f(I_t, p_t) = p_{t+1} \) from pixel intensities \( I \) and current parameter vector to a parameter vector in the next time step, so that a performance measure \( P(I) \) is maximized. The aperture cannot be changed on our camera, see Fig. 2, so \( p = (e, g) \) with exposure \( e \) and gain \( g \). There is a significant variable temporal delay between setting new parameters and an observable change in the output. In the context of Reinforcement Learning (RL) we are using a greedy on-policy immediate reward policy gradient method. The idea is to use little prior information on the problem structure and let the system explore sensorimotor (state-action) space to support autonomous learning. The goal is to find an adequate control policy for stationary illumination conditions \( L \). Figure 3 is an example of a transfer surface in sensorimotor space based on a two-dimensional command value sweep over \( e \) and \( g \) for a single environmental luminance. In other words, resulting image brightness is a function of luminance, exposure and gain and only the latter two are controllable.

The network architecture we are using in our approach has been proposed by Jaeger (2001) and Maass et al. (2002). These models can be subsumed under Reservoir Computing (RC). Reservoirs are randomly and sparsely connected recurrent neural networks which can be tapped by readout units. The reservoir can be excited by external input, readout unit feedback or spontaneous internal activity. The classical approach to training such a network for a particular task is to use regression to find the synaptic weights for the connections from reservoir neurons to the readout neuron. A related concept is that of Extreme Learning Machines (ELM) (Huang et al., 2004) which essentially comprise of a single hidden layer feedforward network randomly connected to input units. The hidden units project onto readout neurons very much like in the reservoir approach.

In an online learning setup, the readout unit weights can also be acquired via error driven or reward driven learning rules (Sussillo and Abbott, 2009; Legenstein et al., 2010; Hoerzer et al., 2012). Initially proposed in the RC context, these methods can straightforwardly be applied to ELMs or Radial Basis Function (RBF) networks. The latter can be pretrained through the Growing Neural Gas (GNG) algorithm (Fritzke, 1995).
where \(|h|\) is the number of histogram bins and the \(a_k\) map the bin index into the pixel value range, \(z_{n,t}\) denotes the \(n\)-th readout signal or policy component and \(m_{\text{target}}\) is the desired mean pixel value. The mean pixel value is not necessarily the best measure of image quality but it is trivial to compute, works reasonably well in all experimental settings considered and it can be swapped for any more elaborate measure without modifying the rest of the system. The state update is then computed by

\[
\Delta x_t = \lambda W_{\text{res}} r_t + W_{\text{inp}} u_t \\
x_{t+1} = (1 - \tau) x_t + \tau \Delta x_t
\]

with state vector \(x_t\), scalar gain \(\lambda\) and leak rate \(\tau = 0.2\). The activation vector \(r_t = \tanh(x_t)\) and \(W_{\text{res\_im}}\) denote the reservoir and input weight matrices respectively. The entries in both matrices are drawn from uniform distributions. In addition, we set 90\% of all the entries in the reservoir matrix to zero. Note that the input vector \(u_t\) also contains the readout signals from the last time step. This notation deviates slightly from the usual one. We form scalar readout units \(\hat{z}_i\) by letting

\[
\hat{z}_{i,t} = \tanh(u_{i,t} r_t)
\]

The components \(w_{i,t}\) of the readout weight vector \(w_{i,t}\) are the plastic elements and subject to modification during the learning process. The actual performance measure \(P_t\) differs for some of the experiments and will be given in the relevant sections. The modulator \(M_t\) is uniformly defined as

\[
M_t = \begin{cases} 
1 & \text{if } P_t > \tilde{P}_t \\
0 & \text{otherwise}
\end{cases}
\]

where \(\tilde{P}_t\) is a low-pass filtered version of \(P_t\). It is computed by

\[
\tilde{P}_t = (1 - \alpha) \tilde{P}_{t-1} + \alpha P_t
\]

with \(\alpha = 0.2\). The effect of the modulator is to allow plastic changes only when the performance has recently improved. During learning, exploration noise is applied to the readout via

\[
z_{i,t} = \hat{z}_{i,t} + \nu_{i,t}
\]

before the readout signal is sent as a command to the camera. The exploration noise \(\nu_{i,t}\) is drawn from a zero-mean Gaussian distribution. For being able to escape saturated (vanishing gradient) regions of the transfer curve, we scale the exploration noise amplitude adaptively by incrementing or decrementing it by a small constant depending on whether \(P_t\) passes a lower threshold. The weight update is performed by determining

\[
\Delta w_{i,t} = \eta_t r_t (z_{i,t} - \hat{z}_{i,t}) M_t
\]

where \(\tilde{z}_{i,t}\) is a low-pass filtered version of the \(i\)-th readout signal and \(\eta_t\) is the corresponding learning rate. This is the Exploratory Hebbian (EH) rule proposed by Legenstein et al. (2010) and modified for use with binary rewards by Hoerzer et al. (2012). The EH rule works by amplifying correlations between exploration noise and the derivative of the performance measure. We are using an exponentially decaying \(\eta\) in all of our experiments. In contrast to the systems considered in Hoerzer et al. (2012), we are faced intrinsic response latency \(T\) which results from response times of the camera and interaction with the camera driver. The command transfer and image acquisition processes are separate and asynchronous, so we introduce a fixed delay on the order of 1 - 2 frames, at a frame rate of 10 Hz, to wait for the image to settle at the current command settings before the response is evaluated through the performance function. This delay corresponds to the maximum delay that occurred. Thus there is no need to consider \(T\) explicitly in the weight update rule (7).

3. EXPERIMENTS

All experiments are performed on the target platform, a quadrotor with an onboard camera (Sony PS3 Eye) and computing hardware. The exposure command \(e_t\) is an integer in the interval \([0, 255]\), the gain command \(g_t \in [0, 63]\), respectively. These are mapped into the \([-1, 1]\) interval for use with the tanh neurons when crossing the network’s boundaries. Additional global scaling is applied for the reservoir inputs by multiplying the input weight matrix \(W_{\text{inp}}\) with a scalar gain. The readout weights \(w\) were set to zero prior to learning in all experiments as suggested in Sussillo and Abbott (2009). In the first two experiments, the system is controlled using direct values. That means we directly set the camera controls to a value in the allowed range. In the third experiments, the network output is considered as a rate of change and thus added to the current command value. The overall structure of the learning system common to all experiments is schematically given in Fig. 4.

![Fig. 4. Principal setup of the learning system with the neural expansion layer in the middle (reservoir), readout units to the right, the camera and the performance measure circuits. Input/output scaling operations are omitted for clarity. The lines labelled \(z_1\) and \(z_2\) carry the signals produced by the readout units, which map onto to exposure and gain controls in the camera, \(I\) denotes the pixel intensity matrix and \(u_i\) is the reservoir input vector, constructed by concatenating external input and efferent copies of the motor signals.](image-url)

3.1 Training a single readout

In this experiment we train a single readout unit to only control exposure. The camera gain is clamped to a
suitable value. Suitable means setting gain to such a value, that varying exposure actually produces a varying sensor response. If for example, ambient brightness is low and gain is set to a low value, varying exposure might not have any observable effect. This is another way of saying that exposure has considerably less effect on pixel brightness than gain. An exemplary run of this configuration is plotted in Fig. 5. The caption text provides some details on this run. The performance measure here is defined as

$$P_t = -|m_{\text{target}} - m_t|$$

(8)

We do not need to use a quadratic error measure as the error is reduced to a binary signal in the modulator. In the optimal case performance $P_t$ is close to zero.

Fig. 5. Training a single readout controlling exposure only while gain is clamped to $g = 20$ with a reservoir of size $N = 20$. Exploration noise $\nu$ is applied during the entire episode. The top panel displays a trace of the complete reservoir activation. The onset of learning is delayed until the reservoir has settled and the $P_t$ transient has vanished at $t = 25$ which is visible in the mid panel. Weight changes quickly drive the system towards maximum reward, until the “knee” is reached at $t = 50$ where learning has overshot. This is corrected in the ensuing phase. The red line $|w_1|$ traces the weight vector norm, $c$ is a scaling constant for the plot. In the bottom panel the two readout activations $z_1$, mapped to exposure, and $z_2$, mapped to gain, are plotted.

3.2 Training two readouts simultaneously

Here we use two readouts which need to cooperate to set the overall sensitivity to a suitable value. We proceed to train two readouts concurrently giving only a single common reward to both plastic processes. The performance function is identical to (8). In this specific case, the learning system needs to discriminate correlation between readout activity and performance evolution for two motor commands which both effect the same variable, mean pixel value. Gain acts much more strongly on the response and sometimes ends up dominating the process. Due to the way exposure and gain interact, the solution now becomes ambiguous. Particularly interesting solutions, sometimes found during learning are oscillatory modes which we observed in different phase configurations, both in-phase oscillation of the two controls as well as counter-phase oscillations. Two example runs with reservoirs of size $N = 20$ are shown in Figures 6 and 7.

Fig. 6. Two readouts, $z_1$ and $z_2$, connected to exposure and gain controls being trained simultaneously. The onset of learning is delayed as in the single readout case. Here, reward takes longer to start climbing. Gain ($z_2$) err off into the wrong direction. At $t = 100$ the system transitions through an oscillatory phase and settles in a suboptimal reward region. There are still some oscillations visible in the activation trace at the top resulting from intrinsic reservoir dynamics.

Fig. 7. Another run of simultaneously training two readouts. Here gain clearly dominates as can be seen both in the middle and bottom pane. The downward spikes in the reward curve are artefacts from artificial lighting.

3.3 Training two readouts in differential value mode

For this experiment we first let the reservoir connection matrix $W_{\text{res}} = 0$, thus removing any recurrence in the expansion layer. The setup now corresponds to an ELM, although there are still two global feeback paths via the efferent copies. Then, we use differential commands, that is, we interpret the readout signal as an increment to the command value set at $t - 1$. In the case of the exposure path this is effected by letting

$$e_{t+1} = e_t + z_{1,t},$$

(9)

and analogously for gain. Differential command coding greatly amplifies the generalizing capacity of the trained network. The direct value experiments above realize a static mapping for a stationary ambient brightness setting. The differential values work across a range of similar situations which only differ in the absolute values that the sensor state variables take on. For this learning task, the
performance measure was changed to
\[ P_t = -|m_{\text{target}} - m_t| + |m_{\text{target}} - m_{t-1}| \quad (10) \]
in order to generate a reward based on the error derivative. An exemplary run is given in Fig. 8.

![Fig. 8. Two readouts, \( z_1 \) and \( z_2 \), connected to exposure and gain controls being trained simultaneously on a feed-forward non-linearly expanded input. The system oscillates for a short while but stabilizes quickly, after about 10 seconds, near the desired quality measure \( m_{\text{target}} \). Note the different y-axes in this plot as compared to the direct value experiments. Quality is the image brightness measure \( m_t \) scaled into the \([-1, 1]\) interval, Commands are the actual command values passed to the camera driver and readouts plots the pure readout signals which are interpreted as an increment to the currently active command value.](image)

### 4. RESULTS

#### 4.1 Statistics

Statistics over 100 runs for the direct value dual readout configuration with reservoir size \( N = 50 \) result in the following, see also Fig. 9: In 77% of runs, the system reaches an error above -10, measured as the average reward \( P_t \) over 30 time steps with zero representing the optimum. The measured errors in the untrained and trained cases are here called “err”. The error in converged state is computed as the average error over a window of size 30. The average time of convergence over these cases is 133 time-steps including the delayed learning onset. The unsuccessful cases are characterized either by premature saturation of the gain value resulting in an unresponsive system or by strong oscillations. This suggests that the learning rate is too large in magnitude.

Similar statistics over 100 learning runs for the differential value dual readout configuration result in a similar picture. Here we also compute the mean absolute error for the untrained and trained state of the network. In 80% of runs the system converges to a final error of more than -12.8 as can be seen in Fig. 10. The initial error and the respective time to convergence are measured while exploration noise is still applied. The numbers for the error in converged state are determined in the final testing phase of the run without exploration noise. The testing phase is clearly visible starting at \( t = 450 \) in Fig. 8.

In summary, the learning system finds an adequate solution in the majority of cases without having information about the sign of the error or the sign of the derivative of the state variables with respect to the motor commands. Special measures could be taken to restart learning in the failing cases by tracking the performance measure and detecting stalling trends. In our particular setup, the image brightness controller is acquired within 10 seconds on average. Given that changing environmental illuminance is often changing slowly and gradually, long-term adaptation could be realized by modulating the learning rate.

![Fig. 9. Time-to-convergence \( t_c \), converged error \( \text{err}_c \) and initial error \( \text{err}_i \) in untrained state plotted for 77 successful out of 100 total runs. Time-to-convergence \( t_c \) is measured by finding the first occurrence in the run of a window with an error below a threshold. The fluctuations in the \( \text{err}_i \) curve are due to changing external lighting conditions. The short runs 5 - 25 are due to initially optimal conditions.](image)

![Fig. 10. Run statistics for the differential command case showing time-to-convergence \( t_c \), converged error \( \text{err}_c \) and initial error \( \text{err}_i \) plotted for 80 successful out of 100 total runs.](image)

#### 4.2 Comparison with other approaches

Direct comparison is difficult because of the structural differences induced by different approaches. It has to be kept in mind that no prior information about the problem is engineered into the learning setup. We tried to maintain this generality in the comparison. Standard supervised learning thus is not applicable as the target output is not known a priori. One possible approach would be to train a forward / inverse model pair (Dearden, 2008) using regression on an ELM or reservoir. The drawback is that exploration is done blindly and incoming information is not immediately used to improve the performance.
We did compare our system to online table-based Q-learning using a coarsely discretized state-action space. In this experiment, we used differential command values, which also reduces the size of the action space. Gain was held fixed and only shutter time was varied. Using an episode length of 10 time-steps, the Q-learning setup takes about 400 episodes to converge to a solution with a performance comparable to that of the reservoir setup. This is an order of magnitude slower than our approach.

We also applied the same learning-rule to a nonlinear expansion through a Radial Basis Functions (RBF) network. The RBF network was pretrained with the Growing Neural Gas (GNG) Fritzke (1995) algorithm to represent sensorimotor space with ten nodes. The initial exploration is again done blindly. In the second learning phase we train the weights from the RBF units to the readout unit with the same learning rule as in our earlier approach. The time to convergence is of the same order as that of the reservoir training as is shown exemplarily in Fig. 11 suggesting that a problem specific tuning of the expansion layer does not provide a significant advantage for this simple problem.

Future work will involve more comprehensive consideration of the issues of representation, temporal delay and refinement of the reward functions. The performance measure we used here is not necessarily the best one, but it is trivial to substitute it with any other. As is the case for network input calibration and scaling, statistical moment based measures and feedback from subsequent processing steps could be integrated into reward computation. The issue of temporal delays in sensor response is related to the Temporal Credit Assignment problem in classical RL. For increasing autonomy, the learning of temporal delays needs to be taken into account. A multiple model or correlation based approach could be considered for this task.

5. CONCLUSION

Fig. 11. The upper pane displays the GNG graph after completion of the node placement learning phase. These are the node centers used for readout training in the learning phase depicted in the lower graph. The isolated episode of activity around $t = 250$ results from perturbation by camera motion and ensuing threshold crossing of the performance value $P$, which in that case controlled learning activity.

We presented an embodied model system of robotic sensorimotor exploration for motor learning by applying a learning rule to learn linear combinations of activations of a nonlinear input expansion layer. Our system acquires a control policy which stabilizes a goal variable within a target range autonomously. It is autonomous in the sense that no indication is given to the learning system about target range autonomously. It is autonomous in the sense that no indication is given to the learning system about the functional relationship between motor commands and sensory effect. Sensorimotor space is explored on-policy in a goal-directed manner in a local neighbourhood of the state trajectory. The system works reliably under diverse conditions in a real world setting and is thus ideally suited for vision-based navigation of MAVs in unprepared environments.

We thank the KoRo group members for inspiration and discussion and W. Maass for pointing out to us learning rule based reservoir training. This work was funded by the DFG Research Training Group METRIK.

REFERENCES