Machine Learning Methods for State Tracking and Optimal Control of Production Processes

Norbert Link and Johannes Dornheim

Intelligent Systems Research Group, Karlsruhe University of Applied Sciences
norbert.link@hs-karlsruhe.de, johannes.dornheim@hs-karlsruhe.de

Abstract. Manufacturing processes act on workpieces by exerting a sequence of varying control actions. This results in a sequence of inner and outer workpiece states. The goal is to reach a final state, which has dedicated geometrical and physical properties. Variations of the input and stochastic influences must therefore be compensated during processing, while the ressource-efficiency should be maximized. For this purpose, self-optimizing Artificial Intelligence (AI) control methods were developed. The corresponding Markov Decision Problem is solved via Machine Learning methods. The cost trade-off between pre-production data sampling to learn the required models and initial low-quality production with learning from production-experience is addressed by two corresponding approaches. 1) Deep Neural auto-encoders and state trackers deliver the input of an optimizing process control, which is constructed from Approximate Dynamic Programming with integrated Neural Networks, representing the learned process dynamics. 2) An explorative AI approach with re-inforcement learning, which automatically learns an implicite model for the control policy, based on the experience with each processing result. This approach can also adapt to process drifts (e.g. from tool wear). Other than classical control methods such as Model Predictive Control, the new approaches can compensate input quality variations, stochastic state perturbations and slowly varying conditions.

Keywords: intelligent control, approximative dynamic programming, reinforcement learning

1 Introduction

The series production of parts is a repetition of processes, transforming each part from an initial state to some desired end state. Each process is executed by a finite, discrete or continuous sequence (*path*) of -in general irreversible- processing steps. Examples are plastic deformations in forming processes or material removal in milling. The optimal control of such processes is different from classical control theory problems and self-learning technologies have to come into place to adaptively solve the optimization problem under varying conditions. The process path optimization is formulated as a Markov Decision Problem (MDP) with finite horizon. The optimization strategy (*policy*) can be refined between subsequent processes, which are called *episodes* in the field of Machine Learning. This allows the adaptation of the strategy to changing process conditions, e.g. to varying state transition functions.

The episodic fixed-horizon manufacturing processes considered here are nonlinear stochastic processes. Every episode in the process consists of T irreversible control steps at discrete times. Deep drawing is used as sample process throughout this paper and is depicted in fig. 1. Based on the measured quality of the process episode result, costs are

assigned and transfered to the control agent by a reward signal R_T at the end of each execution. The control effort (energy consumption, material use, processing time, etc.) is reflected by intermediate costs and also being subject to optimization. The goal is to find a control policy, that minimizes the total cost and thereby optimizes the process performance regarding the resulting product quality and the process effciency. The multistage optimization problem of our interest is described by a MDP and is solved by Dynamic Programming (DP) or Approximate Dynamic Programming (ADP) or by Reinforcement Learning. The DP/ADP/RL approaches in this paper involve the following concepts:

- The value or cost or reward function $J_t(\mathbf{x}_t)$ that defines the optimal value of the optimization objective (defined over the remaining time horizon) for the state $\mathbf{x}_t$ at time t
- In a Markov Decision Process, a transition from a precedessor state $\mathbf{x}_t$ to a successor state $\mathbf{x}_{t+1}$ taking the decision or action u_t is subject to uncertainty and can be expressed by the conditional probability $P(\mathbf{x}_{t+1}|\mathbf{x}_t, u_t)$.
- The control policy π_t that maps the current state $\mathbf{x}_t$ to the optimal action $u_{t_{opt}} = \pi_t(\mathbf{x}_t)$.

The optimization tries to select a control action, which leads to minimum expected cost in the ongoing process. It therefore requires a prediction model, which is built via machine learning and applied via AI inference. The individual model components (costs, state transition, Q-function and policy) are represented by continuous ANNs or by discrete value tables in combination with a k-Nearest Neighbor (k-NN) predictor. The mapping between the actual state $\mathbf{x}_t$ and the (optimal) action $u_{t_{opt}}$ is referred to as the (optimal) policy (i.e., control law). The decision is made based on the stagewise costs $C_t(\mathbf{x}_t, u_t)$ which depend on the selected state transition. To optimize the entire process consisting of multiple time stages, the total costs $\langle \sum_{t=1}^{T} \gamma_t C_t(\mathbf{x}_t, u_t) \rangle$ need to be considered, where γ_t is a discount factor over time ($0 < \gamma_t \leq 1$). The result of the optimization defines the optimal decision for each considered state at every point in time. According to Bellman's Principle of Optimality [4], the multistage optimization problem can be reformulated in terms of the Bellman Equation [5]:

$$J_t(\mathbf{x}_t) = \min_{u_t \in U_t} \{C_t(\mathbf{x}_t, u_t) + \gamma_t \langle J_{t+1}(\mathbf{x}_{t+1}) \rangle\} \tag{1}$$

with $\langle J_{t+1}(\mathbf{x}_{t+1}) \rangle = \sum_{\mathbf{x}_{t+1} \in X_{t+1})} P(\mathbf{x}_{t+1}|\mathbf{x}_t, u_t) J_{t+1}(\mathbf{x}_{t+1})$.

J_t describes the costs that are associated with the current state $\mathbf{x}_t$, and J_{t+1} stands for the costs that are associated with the remaining time steps and depends on the successor state $\mathbf{x}_{t+1}$ of the current decision. Since the state transition is subject to uncertainty, the expectation of J_{t+1} is taken with respect to all possible successor states X_{t+1} and their conditional probabilities $P(\mathbf{x}_{t+1}|\mathbf{x}_t, u_t)$. Two approaches are possible: The first learns explicit initial models of the process state dynamics and of the process state measurement in advance, based on experimental data from laboratory or simulation. The second starts building an optimization model from scratch during processing, learning directly from process data in a trial-and-error fashion. In the first case, much experimental and modelling effort is required, before production can start. In the second case, the initial production will be sub-optimal until the models are sufficiently learned. Nevertheless both AI approaches result in highly optimized production strategies, which can both compensate stochastic influences, while the second can also automatically adapt to changing input quality, to tool wear and other influences, thus maintaining continuous high product quality.

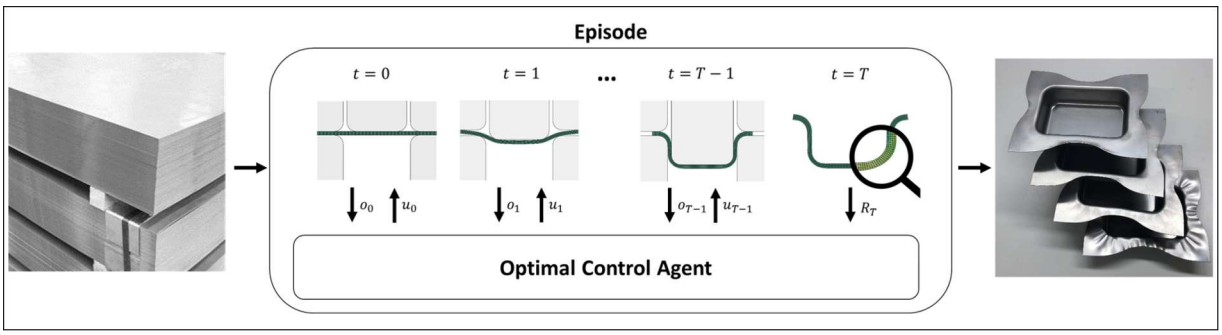

Fig. 1. Deep drawing optimal control as episodic fixed-horizon process. For every control step t, the optimal control task is to determine the control-action u_t that maximizes the expected episode reward R_T, based on the observations $[o_0, ..., o_t]$ in the current episode and on data from previous episodes. (source: [1])

We use the optimal control of a deep drawing process with forced punch speed as application example and evaluation case. In cup deep drawing, a metal sheet is clamped between a blank holder and a die. A punch presses the sheet into the die such that we obtain a cup-shaped workpiece. If the blank holder force is chosen too high, material cracking will occur, while a too small blank holder force will lead to wrinkles in the sheet edge. This necessitates the control of the blank holder force during the execution of the deep drawing process for a finite time horizon. Besides given input properties, like the initial blank shape and thickness variation, the time-variation of the blank holder forces is crucial for the resulting process quality. The control goal is to set the blank holder force at each processing step, depending on the current, partially observable process state in a way, which yields the optimal process result. The influence of space and time variation schemes of blank holder forces on the process results is examined e.g. in [2], [3] and [4]. The state of a workpiece during processing is described by the stress distribution within the material, but which is not directly observable during processing. Only the reaction force in the punch and the intake of the material between die and blankholder can be measured. This makes the optimization a so-called *partially observable Markov decision problem* (POMDP).

2 Optimal control for continuous state spaces by incremental learning

The first approach presented in [5] uses model-based optimal control methods where a process model, an state-observation model and a cost function are available from previous work [6]. Optimal control is then acheived by dynamic programming [7]. Such approaches are subject to the so-called curse of dimensionality in high-dimensional state spaces, leading to diffculties from huge sample sizes required for modelling and from the resulting computational complexity. In the case of continuous (and thus infinite-dimensional) state spaces, the optimal control solution by dynamic programming requires discretization of the state space, leading to suboptimal solutions. These problems are addressed in the field of approximate dynamic programming, combining dynamic programming with function approximation [8]. In applying ADP to the problem, Artificial Neural Networks (ANNs) are created as global parametric function approximators to represent the value functions as well as the state transitions. For each time step of the finite time horizon, time-indexed function approximations are built. We use a backward ADP approach with batch learning

of the ANNs. Here, the ANNs are trained from temporary value tables constructed by exhaustive search backwards in time. The obtained value function approximations are obtained with good performance, where the models for the state transition and the value function are determined with batch learning ANNs from simulation data. The control policy is given by the solution of the stochastic Bellman Equation with approximated successor costs $\tilde{J}_{t+1}$ with respect to the pre-decision state $\tilde{\mathbf{x}}_{t+1}$) (Eq. 2).

$$J_{t_{opt}}(\mathbf{x}_t) = \min_{u_t \in U_t} \{C_t(\mathbf{x}_t, u_t) + \langle \tilde{J}_{t+1}(\tilde{\mathbf{x}}_{t+1}) \rangle\} \tag{2}$$

The optimization is performed backwards in time as depicted in Fig. 2.

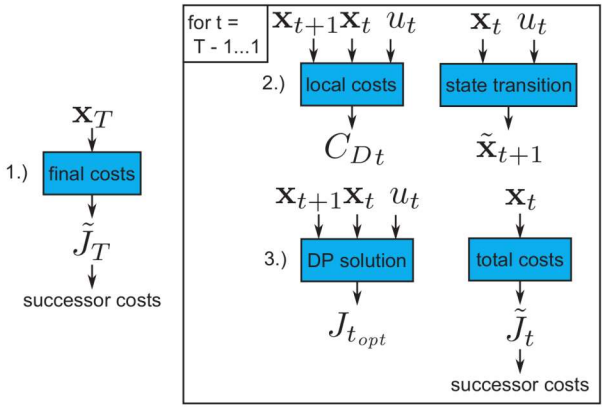

Fig. 2. Scheme of backward ADP (source: [5])

We start by determining the final costs $J_T = C_T = C_F$ for each final state $\mathbf{x}_T$ and create the approximation $\tilde{J}_T$ in terms of a batch learning ANN. This serves as the successor costs $\tilde{J}_{t+1}$ in the first loop run for $t = T-1$. For each loop run, we determine the local costs C_t and create a batch learning ANN to predict the state transition from actual states $\mathbf{x}_t$ with decisions u_t to the corresponding successor states $\tilde{\mathbf{x}}_{t+1}$. In a next step, the Bellman Equation (with approximated successor costs $\tilde{J}_{t+1}$ is solved with respect to the pre-decision state

$$J_{topt}(\mathbf{x}_t) = \max_{u_t \in U_t} \{C_t(\mathbf{x}_t, u_t) + \langle \tilde{J}_{t+1}(\tilde{\mathbf{x}}_{t+1}) \rangle\} \tag{3}$$

for each state $\mathbf{x}_t$ and the optimal solution J_{t_opt} is stored in a temporary value table. The approximated successor costs $\tilde{J}_{t+1}$ are either given as an ANN from the previous loop run or by the approximated final costs $\tilde{J}_T$ at the beginning. The expectation $\langle \tilde{J}_{t+1} \rangle$ is calculated as a numerical integral. The content of the temporary value table then serves for training a batch learning ANN $\tilde{f}_t$ based on the states $\mathbf{x}_t$. This approximation corresponds to the successor costs in the following loop run. The Backward ADP gives an optimal value-function from which the optimal control strategy can be derived, but which is not adaptive to process drifts.

3 Reinforcement Learning Approach

The processes considered here are not fully observable. The sole sensor observations at a time are not suffcient for deriving optimal control decisions. Instead of learning measurement models (mappings of sensor data history on states) as in the previous

approach, the accumulated sensor data and control history is taken into account to avoid any pre-production modelling.

The model-free adaptive approach, proposed in [1], requires no prior information, (like reference trajectories, a process model or an observation model), since the optimal control strategy is learned during execution. The approach can thereby be used, if no accurate process model is available or the use of the given process model for optimization is impractical.

In this approach, following the standard notation of reinforcement learning, the system and the optimization problem are modeled as an MDP, as introduced in the previous chapter. In MDPs, instead of the cost-function J, a reward function R is used, leading to a maximization problem instead of the minimization in eq. 3. The Bellman equation is then given by the value function

$$V^*(\mathbf{x}) = \max_{u \in U} \mathbb{E}_P \Big[R_u(\mathbf{x}, \mathbf{x}_{t+1}) + \gamma V^*(\mathbf{x}_{t+1}) \Big], \tag{4}$$

where the probability of $\mathbf{x}_{t+1}$ is given by the transition probability function $P_u(\mathbf{x}, \mathbf{x}_{t+1})$, capturing stochastic process conditions. The optimization is achieved by selecting the action, which leads to the state that maximizes the value function V^*. This would require knowledge of the state transition probability function. Instead, we use the optimal Q-function, assigning an expected reward value to each state-action tuple

$$Q^*(\mathbf{x}, u) = \mathbb{E}_P \Big[R_u(\mathbf{x}, \mathbf{x}_{t+1}) + \gamma \max_{u_{t+1} \in U} Q^*(\mathbf{x}_{t+1}, u_{t+1}) \Big]. \tag{5}$$

For discrete action-spaces, the optimal policy can then be determined from the Q-function simply by

$$\pi^*(\mathbf{x}) = \arg\max_{u \in U} Q^*(\mathbf{x}, u). \tag{6}$$

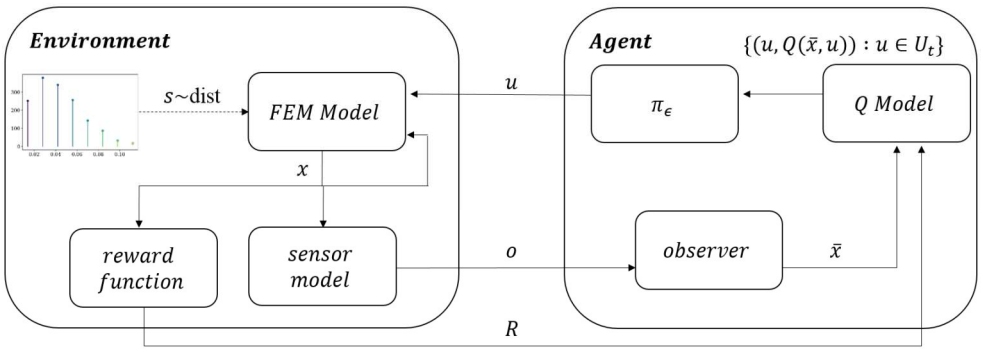

Fig. 3. Scheme of the interaction of the optimal online control agent with the process environment. (source: [1])

By taking the actions into account, the Q-function implicitly captures the system dynamics, and no additional system model is needed for optimal control. A generic version of the Q-learning control agent is depicted in fig. 3.

In Q-learning-based algorithms, Q^* is found by constantly updating a Q-approximation by the update step in eq. 7, using experience tuples $(\mathbf{x}, u, \mathbf{x}_{t+1}, R)$ and a given learning rate $\alpha \in [0, 1]$, while interacting with the process in an explorative manner.

$$Q'(\mathbf{x}, u) = \big(1 - \alpha\big) Q(\mathbf{x}, u) + \alpha \big(R + \gamma \max_{u_{t+1} \in U} Q(\mathbf{x}_{t+1}, u_{t+1}) \big) \tag{7}$$

An observer derives surrogate state descriptions $\bar{\mathbf{x}}$ from the observable values o, and previous control actions u. Control actions are determined based on a policy π, which itself is derived from a Q-function. In the approach, proposed in [1], the Q-function is learned from the processing samples via batch-wise retraining of the respective function approximation, following the incremental variant of the neural fitted Q iteration approach ([9]). For exploration, an ϵ-greedy policy is used, acting randomly in an ϵ-fraction of control actions. To derive the optimal action, the current approximation of the Q-function is used (exploitation). The exploration factor $\epsilon \in [0, 1]$ is decreased over time to improve the optimization convergence and to reduce the number of sub-optimal control trials. We use an exponential decay over the episodes i according to $\epsilon_i = e^{-\lambda i}$, with decay rate λ. Since x is only partially observable, we use the full information about observables and actions for the current episode by concatenating all these values into a surrogate state description $\bar{\mathbf{x}}$. Thus, the dimension of $\bar{\mathbf{x}} \in R^n$ is time-dependent according to $n_t = [\dim(O) + \dim(U)] * t$. When using Q-function approximation, the approximation model input dimension is therefore also dependent on t. If function approximation methods with fixed input dimensions (like standard artificial neural networks) are used, a dedicated model for each control step is required. The extension of the model-free approach for applications with multiple weighted and eventually contrary objectives is described in [10].

4 Results and Conclusion

Both approaches, proposed in [5] and [1], are applied to the standard industrial process of cup deep drawing (soda cans, machine covers, pots, etc.) with volumes of millions of batches per day. The process goal is to produce a cup with low internal stress and low material usage, but with sufficient material thickness. The three optimization criteria are combined into a single Reward value by calculating the weighted harmonic mean of the minimum wall thickness, the negative residual stress and the left-over material. The weight values reflect the emphasis, which is given to the respective goals. The first approach requires requires a high effort in sampling experimental data and training of the models, before it can be applied to the process, but produces nearly optimal results from scratch despite being subject to processing noise. The second approach requires no sampling or training before the start of production. The price is lower performance in the early phase of production, while the models are learnt. It was shown in [1], that in the case of variantions in the friction between material and matrix/blank-holder/punch during processing, which occur due to roughness variations of the sheet material, the second approch learnt to cope with the variations and out-performed the first approach by 20% on the long run.

5 Acknowledgements

The authors would like to thank the DFG and the German Federal Ministry of Education and Research (BMBF) for funding the presented work carried out within the Research Training Group 1483 "Process chains in manufacturing" (DFG) and under grant #03FH061PX5 (BMBF).

References

1. Dornheim, J., Link, N., Gumbsch, P.: Model-free adaptive optimal control of episodic fixed-horizon manufacturing processes using reinforcement learning. arXiv preprint arXiv:1809.06646 (2018)
2. Tommerup, S., Endelt, B.: Experimental verification of a deep drawing tool system for adaptive blank holder pressure distribution. Journal of Materials Processing Technology **212**(11) (2012) 2529–2540
3. Singh, C.P., Agnihotri, G.: Study of Deep Drawing Process Parameters: A Review. International Journal of Scientific and Research Publications **5**(1) (2015) 2250–3153
4. Wifi, a., Mosallam, a.: Some aspects of blank-holder force schemes in deep drawing process. Journal of Achievements in Materials and Manufacturing Engineering **24**(1) (2007) 315–323
5. Senn, M., Link, N., Pollak, J., Lee, J.H.: Reducing the computational effort of optimal process controllers for continuous state spaces by using incremental learning and post-decision state formulations. Journal of Process Control **24**(3) (2014) 133–143
6. Senn, M., Link, N.: Hidden State Observation for Adaptive Process Controls. ADAPTIVE 2010, The Second International Conference on Adaptive and Self-Adaptive Systems and Applications (c) (2010) 52–57
7. Bellman, R.: Dynamic programming (dover books on computer science). (2003)
8. Lee, J.H., Wong, W.: Approximate dynamic programming approach for process control. Journal of Process Control **20**(9) (2010) 1038–1048
9. Riedmiller, M.: Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. In: European Conference on Machine Learning, Springer (2005) 317–328
10. Dornheim, J., Link, N.: Multiobjective reinforcement learning for reconfigurable adaptive optimal control of manufacturing processes. In: 2018 International Symposium on Electronics and Telecommunications (ISETC), IEEE (2018) 1–5