TUM

TECHNISCHE UNIVERSITÄT MÜNCHEN
INSTITUT FÜR INFORMATIK

Teleported AI

Knoll, Alois and Walter, Florian

TUM-I2086

Teleported AI: A Cloud Platform for AI-Based Teleprogramming of Robots

Alois Knoll and Florian Walter

Chair of Robotics, Artificial Intelligence and Real-Time Systems
Department of Informatics
Technical University of Munich

knoll@in.tum.de, walter@in.tum.de

June 2020

Despite recent progress in AI, deploying robots is still a time-consuming and cumbersome process. This is because current robots can only repeat pre-programmed instructions with minimum adaptability to changes. In this report, we introduce the concept of Teleported AI that draws from state-of-the-art deep neural networks, robot simulations and cloud-based software development to enable the massively parallel training of robots in virtual environments. This novel approach alleviates the need for expensive physical robot setups and makes progress in computing technology directly available to robotics with the goal of developing robots with adaptive autonomous behavior through simple direct instruction.

1 Challenge

Programming robots to perform a specific task is still a complex and time-consuming process. It requires specialists, is therefore expensive and only possible if several domain experts communicate with each other. The person who designs *what* is to be built by the robot – the **product engineer** – must communicate with the person who defines *where* (i.e. in which environment and with which equipment) the execution of the robot task (e.g. assembly of a unit) is possible – the **plant engineer**. The latter, in turn, has to talk to the person who will then set up and program the robot, i.e. teach the robot *how to do* this – the actual **robot programmer**.

2 State of the Art

Robots have undergone a considerable development in recent years, towards sensorization (force transducers, cameras), which makes it possible to adapt to changing environmental conditions. If the task remains the same and only e.g. the position of the assembly parts in the robot environment changes, this adaptation already works quite well. Considerable efforts have also been made in terms of mechanical adaptability (tool changers, multi-finger hands).

Nevertheless, the problem of programming robots directly according to the product engineer's idea has not yet been solved satisfactorily: "Do what I mean." It is possible to grasp the robot and "guide" it by applying force, and the robot will reproduce the demonstrated trajectory – this has, by the way, already been possible since the 1970s. However, this does not convey a true *understanding* of the task, which would be required to transfer it independently to another, partly similar task. It is only a basic imitation – possibly with a few correction possibilities if parts are positioned differently than during the learning process.

In conclusion, there are two things missing:

1. The possibility that a robot has an *understanding* of the nature of the task it is performing and thus can collect transferable knowledge for other tasks or even other task types.
2. The possibility of shortening the path from the *product engineer* to the final *robot controller*. Ideally, the product engineer programs the robot directly. This would alleviate the cumbersome communication of his intention across several stages and "middlemen" from this brain into the "brain" of the robot, with each of the stages accounting for information loss, additional inquiries etc.

Both fields of activity explicitly do not refer to the new or further development of mechanics. This reflects that the essential problems in robotics today – as in many other application areas – no longer lie in hardware but in software. In fact, it is the case that algorithmic limitations (e.g. adaptability of a system at runtime) are compensated at great expense by appropriate design of the mechanics and electronics (e.g. high-precision sensor technology and minimal manufacturing tolerances).

3 Opportunities and Limitations of Modern AI

The rapid development of artificial intelligence (AI) in recent years has also opened up new perspectives for robotics. Deep neural networks and deep learning have set a new standard and outperform classical methods in many applications [1]. These also include parts of robotics where the combination of deep neural networks and reinforcement learning has already been successfully applied to a wide range of control problems, including complex sensorimotor coordination tasks, such as grasping objects [2]. This example is already very much in line with the two goals formulated above, since the robot on the one hand can learn the task completely independently and the neural network on the other hand can be trained by means of transfer learning also for related tasks in a similar environment once the initial training has been finished.

A major obstacle for the broader use of deep neural networks in practical applications is – setting aside questions of functional safety – the intense effort for initial training, which requires large amounts of data. In contrast to the classification of images, for example, which are available online in practically unlimited quantities and free of charge [3], training robots requires the collection of data suitable for each task and environment. If data collection is carried out on a physical robot, it is not only expensive, but above all takes time. There are already initial approaches that can be used to reduce the amount of training data and make learning certain tasks relatively quick [4]. However, the increase in efficiency is primarily achieved through manual adjustments to the models, which means that programming for a specific task is again necessary. Simulations are therefore becoming a key tool in robotics, since only in this way can any task of any complexity be learned in any environment. Variations of scenarios can be generated algorithmically without additional effort, thus covering a higher number of situations than would ever be possible in the physical world.

Whether a certain task can be learned from a given data set depends essentially on the learning algorithms used. For the solution of increasingly complex problems, it will, in general, not be sufficient to simply increase the amount of data. Therefore, the further development of the theory of neural networks, or more generally, machine learning and AI, remains central. Realistic brain models, which are becoming increasingly popular in neuroscience, provide promising starting points for this. Simulations can significantly accelerate the development of new AI methods.

In summary, the goal is to achieve the following three robot capabilities using the state of the art in brain research, AI and simulation technology:

1. Robot systems must be endowed with **task intelligence**. This means that a robot must learn directly from the product engineer – i.e. in a dialog with him – how to perform a task. To accomplish this, the product engineer must be able to communicate with the robot in its (human) language. Roles are distributed in the same way as between master and apprentice, although it is clear that there are different terminologies and concepts – a welding robot will communicate in a different way than a watchmaker robot.

2. Robots must have **behavior intelligence**. This means that they must be able to do everything on their own that was not directly taught to them during the initial programming by the product engineer. The knowledge for "bridging" is accumulated during the execution of other tasks. It is obvious that this knowledge, properly abstracted, can then be exchanged via the cloud with other robots that work in the same domain and are in a similar situation.

3. Robots must be endowed with **comprehension intelligence**. If they are guided to complete a task – or receive a command on another communication channel, such as voice – they should be able to understand why that command was given in that particular situation.

4 Technical Foundations

This entire list of demands could not have been met before the availability of cloud access directly on the robot. Today, however, this is the case and robots therefore can access the entire arsenal of methods (abstract, partially instantiated for a class of tasks or completely instantiated for a concrete task) of AI, which is available on cloud servers worldwide. Even more: analogous to autonomous robot taxis, which are planned to be remote controlled by a human beings for a short periods in difficult traffic situations via "teleoperation", a human being can intervene also here in order to help out in difficult situations. This can happen directly during task execution, but also in the programming phase in order to obtain "expert advice" in during the dialog between the product engineer and the robot. This expert does not have to be a human being, but could also be a specialized AI with the necessary knowledge base.

Most of the basic technologies required for the program outlined in this report are readily available in the Neurorobotics Platform (NRP) [5, 6] of the Human Brain Project (HBP) [7], an integrated development environment for the design and execution of robot simulations for applications in neuroscience and AI. It supports cloud services such as Amazon Web Services (AWS), Microsoft Azure or Google Cloud as technical infrastructure. Functionality not yet available in the NRP may need to be added. However, this does not pose a problem since the source code of the system is freely available and can be extended as needed. In particular, the following features and components are required:

1. A **scalable technical infrastructure** for the parallel execution of potentially hundreds of simulations, to accelerate training and to cover as many different problem instances as possible. The NRP has already been tested on various infrastructures such as OpenStack, AWS and Google Cloud. Since offerings for cloud computing resources are largely standardized, other services can also be supported with little effort. It is important that the infrastructure provides suitable accelerators for training and inference in deep neural networks.

2. **Realistic simulation models** of robots and task environments. The quality of the models is crucial to ensure that the virtual training can take place under the most realistic conditions possible. For certain tasks (interaction with objects, materials with specific appearance or special haptic properties), extensions for the simulation may be required to reproduce all characteristics relevant to the training process as realistically as possible. The same applies to certain properties of controllers and sensor systems.

3. A sufficiently large **data storage** that stores simulation data and training results. The required capacity depends strongly on the complexity of the task as well as the amount and resolution of the processed sensor data.

4. Suitable **network architectures, learning algorithms and loss functions** for training. The concrete choice of a particular model depends strongly on the problem to be solved (e.g. camera- or force-guided robot control). However, certain model types usually cover larger problem classes, which means that not every new task also

requires a new model. Nevertheless, the integration of new learning methods should be as simple as possible, so that new architectures can always be implemented and evaluated as required. In the NRP, this has already been successfully implemented in an initial application by defining abstract programming interfaces for a concrete experiment type.

5 Embedding in the Science and Technology Landscape

The rapid progress in the development of deep neural networks would not have been possible without the implementation of an appropriate technical infrastructure. It is not without reason that in recent years many companies have introduced their own platforms for the development of AI systems, ranging from software libraries (Google TensorFlow, Facebook PyTorch, Huawei Mindspore) for the implementation and training of deep neural networks, to corresponding cloud services (AWS SageMaker, Azure Machine Learning), to new processor architectures (Google TPU, Huawei Ascend AI Processor) for the acceleration of training and inference. In addition, the source code of current algorithms and models is usually made freely available on collaborative version management systems (Github, Gitlab).

It is essential for the realization of the agenda proposed here to take advantage of this ecosystem and benefit from the considerable worldwide effort going into the development of models and tools. The fact that solutions are already available for some of the problem areas outlined at the beginning of this report should by no means be seen as a disadvantage. On the contrary, only this way the implementation of ambitious goals becomes possible at all. The actual innovation is therefore not the independent development of a closed system – the multitude of cognitive architectures and system architectures available today gives clear proof of this. The goal is rather the integration and extension of existing solutions for the efficient and timely implementation of the actual use case. Although more and more papers on the application of deep neural networks in AI are being published, there are still hardly any examples of successful practical implementations in productive systems.

The fact that existing tools, algorithms and models are used does not mean that no development work is required. Only in the rarest cases can results – if at all fully documented and published – be directly adopted. Therefore, further development and fine-tuning must be performed in any case.

6 Practical Implementation

As a result of the broad base of published models, the main ingredients for the implementation of simpler tasks (gripping single objects, basic joining operations, adaptive control in case of material scattering etc.) are already available. It is crucial for the first steps that these basic operations are implemented in a robust and flexible way, so that they become available to as many application scenarios as possible. Even the teaching of simple motion primitives can drastically increase efficiency compared to classical programming, if the learned behavior covers a large number of situations (minor changes to the workpieces, repositioning of the robot, changing the tool etc.).

Therefore, a catalog of simple but practice-relevant basic operations needs to be defined first. These operations need to be mastered in a highly efficient and reliable manner. Based on this, increasingly complex scenarios (object manipulation, following an assembly plan, human-robot collaboration, etc.) can then be defined. It seems reasonable to guide the development of this task catalog based on a set of target applications; for example, different kinds production cells in a factory.

The NRP is designed as an open software framework that consistently integrates the components and technologies required for this project. An open system architecture enables the integration of all common tools such as Google, DeepMind, NVIDIA and others. Similarly to Google Colabs, experiments can be configured and managed with Juypter notebooks. In the simulation environment itself, sensors, controllers and models can be added as required.

The implementation of a specific task starts with the definition of a digital twin of the target environment in which all algorithms can then be virtually designed and tested. As mentioned before, a starting point could the modeling of a concrete production cell at an assembly line. This requires not only models and descriptions of the cell itself, but also of the workpieces processed in it. The implementation of the simulation model is critical for all further steps and therefore requires special consideration.

As soon as the simulation in the NRP − the experiment − is fully implemented and validated, the next step is the development of the actual learning algorithms. This can only be done efficiently with rapid design iterations if the experiment can be accelerated through parallel execution. Distributed learning frameworks such as DeepMind impala can serve a basis for the design of an appropriate system architecture. The final deployment to the cloud can be handled by Docker and Kubernetes, both of which have already been tested with the NRP.

As soon as the infrastructure comprised of the simulation models and parallel deployment described above is in place, the actual use cases can be implemented. The completely virtual development process enabled by the NRP enables shorter development cycles compared to working on physical systems and thus also faster evaluation of implementation alternatives.

References

[1] Yann Lecun et al. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539.

[2] Sergey Levine et al. "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection". In: *The International Journal of Robotics Research* 37.4-5 (2017), pp. 421–436. ISSN: 0278-3649. DOI: 10.1177/0278364917710318.

[3] Jia Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[4] Sehoon Ha et al. *Learning to Walk in the Real World with Minimal Human Effort*. 2020. URL: https://arxiv.org/pdf/2002.08550 (visited on 03/10/2020).

[5] Alois Knoll and Marc-Oliver Gewaltig. "Neurorobotics: A strategic pillar of the Human Brain Project". In: *Brain-inspired intelligent robotics: The intersection of robotics and neuroscience*. Ed. by Jean Sanders and Jackie Oberst. Washington, DC: Science/AAAS, 2016, pp. 25–34.

[6] Egidio Falotico et al. "Connecting Artificial Brains to Robots in a Comprehensive Simulation Framework: The Neurorobotics Platform". In: *Frontiers in Neurorobotics* 11 (2017), p. 2. ISSN: 1662-5218. DOI: 10.3389/fnbot.2017.00002. URL: https://www.frontiersin.org/article/10.3389/fnbot.2017.00002.

[7] Katrin Amunts et al. "The Human Brain Project—Synergy between neuroscience, computing, informatics, and brain-inspired technologies". In: *PLOS Biology* 17.7 (2019), e3000344. DOI: 10.1371/journal.pbio.3000344. URL: https://doi.org/10.1371/journal.pbio.3000344.