Künstliche Intelligenz manuscript No.
(will be inserted by the editor)

The Optimization Route to Robotics—and Alternatives

Marc Toussaint · Helge Ritter · Oliver Brock

Received: date / Accepted: date

Abstract Formulating problems rigorously in terms of optimization principles has become a dominating approach in the fields of machine learning and computer vision. However, the systems described in these fields are in some respects different to integrated, modular, and embodied systems, such as the ones we aim to build in robotics. While representing systems via optimality principles is a powerful approach, relying on it as the sole approach to robotics raises substantial challenges. In this article, we take this as a starting point to discuss which ways of representing problems should be best-suited for robotics. We argue that an adequate choice of system representation—e.g. via optimization principles—must allow us to reflect the structure of the problem domain. We discuss system design principles, such as modularity, redundancy, stability, and dynamic processes, and the degree to which they are compatible with the optimization stance or instead point to alternative paradigms in robotics research. This discussion, we hope, will bring attention to this important and often ignored system-level issue in the context of robotics research.

We thank the German Research Foundation for the creation of the Priority Programme SPP 1527, by which this research is supported.

M. Toussaint
Machine Learning & Robotics Lab, U Stuttgart
E-mail: marc.toussaint@informatik.uni-stuttgart.de

H. Ritter
Neuroinformatics Group, U Bielefeld
E-mail: helge@techfak.uni-bielefeld.de

O. Brock
Robotics and Biology Laboratory, TU Berlin
E-mail: oliver.brock@tu-berlin.de

1 Introduction

Robotics is making progress in huge strides, some people say. Others disagree and believe robotics has not been able to establish itself as a discipline distinct from control, mechanical engineering, and computer science. This disagreement should be reason enough for the robotics community to ask some questions.

A curious look over to some other rather young disciplines may prove helpful. Take machine learning, for example. Originally a part of computer science and AI, it now has established itself as a new discipline with whole university departments dedicated to it. One of the catalysts of progress in machine learning has been the extensive use of optimization to formalize problems. In fact, large parts of modern machine learning are covered when referring to one of the many supervised or unsupervised learning objectives (losses, regularizations, embedding costs, KL-minimization, etc). A similar situation is found in the field of modern computer vision, which largely builds on rigorous mathematical programming formulations, often suited for GPU solvers.

This raises the question of why the use of optimality principles has been so successful in these disciplines. And whether robotics can benefit in similar ways from

a wide-spread adaptation of optimization, beyond its already established use.

To[1] *answer these questions we believe it is essential to expand the discussion to include a more fundamental level. We take the stance that the use of optimization principles is merely a means to represent systems (as explained in detail in the next section). This choice of system representation is helpful if and only if it reflects the structure of the problem domain. This is analogous to machine learning, where a good representation must reflect the structure of the underlying data distribution, thus encoding a suitable prior to achieve generalization. On the other hand, in contrast to machine learning, we now apply this insight to evaluate the suitability of a representation of robotic systems, instead of data.*

In this paper, we therefore aim to discuss the use of optimality principles in view of whether this allows us to reflect the structure of robotics problems in a promising way. We believe that robotic systems are in some respects different to typical machine learning or computer vision "systems". A *robotic system* is an embodied entity that couples internal computational processes with external processes (in the sense of *ecology*, as introduced by the psychologist James J. Gibson) to generate task-directed behavior. These internal computational processes are of particular concern here and could in principle be described using optimality principles. Whether or not doing so promises the same progress as in other disciplines will be examined in more detail in the remainder of this paper.

Our answer to this question is ambivalent. We think that robotics can benefit from following even more the examples of machine learning and computer vision and from describing integrated behavioral systems and problems via rigorous optimality principles. But at the same time, we will point to reasons that integrated robotics problems are structurally different to machine learning (ML) and computer vision (CV) problems. These differences, we will argue, necessitate design principles that may be difficult to capture in the context of optimization formulations.

The ambivalence towards optimization opens up the question: Are there other formalisms or principles that are better suited for the development of robotic systems? Because they more aptly capture the inherent structure that arises from coupling internal and external processes? We will also explore this question, proposing alternatives and considering their suitability.

In the following section, we briefly elaborate on optimization as a means of system representation. Section 3 collects some example areas within robotics where optimization approaches are well-established and led to great progress, but also where they are questionable. Section 4 aims to distill what might be a major issue with describing robotic systems *holistically* via optimality principles. Section 5 discusses alternative design and representation principles of robotic systems. Section 6 then considers the view that we should discuss the *computational processes* used, of which optimization processes are just one (special) example next to others.

Finally, section 7 will provide three separate conclusions from the individual authors, offering personal perspectives on the *Optimization Route to Robotics* and future research.

2 Optimization as a means of modeling systems

To appreciate the idea of using optimality principles to represent systems, we take a brief digression into physics. The path of a particle can be modeled as the extremum of an action integral. Clearly, this is not a normative statement about particles, or about the "motivation" of particles. It is a mathematical tool to concisely describe paths, simply an *alternative* to other ways of describing it, such as a differential equation.

In other disciplines, psychologists or sociologists describe the behavior of humans and whole societies as the extremum of some criteria. Again, the normative interpretation, or the popular conclusion that humans actively seek to "optimize" these criteria is misleading. They just behave in a manner that can be modeled as extremum to an objective. This shows that "being an optimum" is entirely unrelated "being good" or "bad". *Optimization is simply a method to describe systems.*

Can every system—and therefore also robotic systems—be described in terms of optimality principles? Clearly, yes! Feynman already pointed out that any given set of equations (including all equations of physics) can be (trivially) rewritten into a single large optimization function (Feynman called its value the "unworldliness measure") with a single global minimum exactly when all equations are fulfilled (Feynman et al., 1963).

[1] With this formatting we will mark text that relates the discussion back to the overarching argumentative flow.

Whether optimization principles are a good choice to describe a robotic system is a very interesting question. Why have optimality principles been such a great tool to describe natural phenomena, particularly on the physical level? In the introduction we mentioned that one should judge any choice of representation based on whether it allows us to capture the problem structure. Can we identify the type of structure that can well be reflected in formalisms for optimization? And thereby identify the category of problems for which optimization will be successful?

As in physics, optimality principles have become a central means of describing systems in the context of machine learning. This trend was a progression from system modeling based on explicit rules, such as Hebb's rule, again analogously to differential equations in physics. Today, most learning systems are described using objective functions. In fact, the discovery of suitable objectives is a major outcome of machine learning research: just as physics discovered discriptions of the path of a particle, machine learning discovered discriptions of what makes a good learner (i.e. what are priors, regularizations and losses that lead to well generalizing learning systems). Such objectives for instance include mixing unsupervised objectives with supervised objectives (semi-supervised learning) to improve the "quality" of internal representations, e.g. as in deep learning Weston et al. (2008). The success of machine learning can be viewed as an indication that optimality principles allow us to adequately capture structure in the types of data prevalent in this domain (in terms of losses, regularizations, priors, etc).

In the next sections we will explore if optimization principles have had—or possibly could have—the same positive impact in the field of robotics. If they allow for a similarly concise description of performing robotic systems. If so, it could imply that these principles also capture structure inherent to problems in robotics. If not, we are faced with the challenge of characterizing the structure of robotic problems and finding alternatives approaches and principles, replacing or complementing optimization as a representational and algorithmic tool for the development of robotic systems.

3 Existing Optimality Approaches in Robotics

Let us briefly collect some examples from robotics where optimization approaches have been particularly successful or seem particularly problematic.

The whole field of control theory (let us include Markov Decision Processes and Reinforcement Learning approaches with this) lays one of the most important foundations of robotics. What would robotics be without inverse kinematics, operational space control, force control, etc.? Each of these can concisely be described in terms of optimality principles[2]—but also just in terms of their laws, of course. However it seems without doubt that in the area of control (and learning/adaptive control) the optimality approach was very successful as a method to develop a large variety of efficient solutions.

Optimization principles have also become a mainstay of robot path generation, a very extensive area of research. While sampling-based approaches, such as Rapidly Exploring Random Trees and Probabilistic Road Maps are the dominant method of choice, the optimization view more recently led to a series of methods (Ratliff et al., 2009; Toussaint, 2014) that are strong in terms of speed in high-dimensional settings, but weak in terms of global completeness, and therefore complementary to sampling-based methods.

Why are optimality principles so apt for these domains? We hypothesize that they can very well reflect core desirable properties of motion—esp. smoothness. In fact, the optimality principles on top of the linear algebra and the non-linear but smooth (Riemannian) geometry (of kinematics and dynamics) and metrics used to describe desirable properties of controlled motion seem to directly reflect the true physical structure of local motion. For global aspects of motion, it seems that optimality criteria are more problematic. This shows once again that adequate ways of representing a problem should reflect the structure inherent to that problem.

Concerning the interaction with contacts (including grasping) the situation is less clear. Early optimization-based approaches to grasping (maximizing wrench closure) were fragile due to unrealistic assumptions about the availability of exact models of hand and object as well as highly precise actuation. In contrast, appro-

[2] See (Laumond et al., 2015) for a non-technical introduction.

priately chosen compliant control leads to significantly more robustness, even when not making these assumptions. More recently, however, there is increasing effort to more rigorously cope with contact interactions also in optimization-based approaches: e.g. utilizing linear complementary problem formulations within optimal control through contacts (Posa et al., 2014) and physical simulation (Mordatch et al., 2012). While these approaches do not (yet?) help for real-world grasping, they do solve problems that would otherwise be inconceivably to solve by hand-designed controllers (e.g. the landing of an airplane on a wire (Moore et al., 2014)).

While we are not experts on walking, in our perception also here the optimization-based approaches make more and more progress—in particular also because of hardware developments that in fact bring the real system closer to the theoretical models, such as direct drives (Seok et al., 2013) and force control with hydraulic actuators.

We already mentioned the deficits of classical optimal closure approaches to grasping. More modern versions of optimality approaches to grasping include alternative measures to evaluate good grasping or caging areas (including one of the authors (Zarubin et al., 2013)), and rigorous POMDP approaches to generate complex grasp behaviors (that include tactile exploration to reduce uncertainties) (Koval et al., 2014). Still, grasping experts would claim that hand-designed grasp strategies that exploit hand morphology and compliance largely outperform other approaches. One reason is the lack of a precise model (including model uncertainties) of the object and material to be grasped; a second that even if we had a precise model, using it to compute optimal grasp behaviors is computationally complex and would typically not lead to robustness against model errors, control uncertainties or unexpected environment dynamics.

4 A potential core problem: decentralization & modularity

It is interesting to note that the above examples of success all concerned *isolated* sub-problems of robotics. In contrast, it seems particularly hard to describe integrated systems in terms of a single overall optimality principles. The work by one of the authors (Toussaint, 2009) proposes an integrated architecture, describing all system aspects in terms of a single objective function (more precisely, probabilistic inference). Such approaches suggest that, in fact, integration might be simpler once all components have consistently been formulated in terms of optimality principles as this is "just" a problem of combining mathematical programs—instead of a matter of software engineering. However, the practical challenges are enormous and the state-of-the-art in robotic systems clearly does not point towards such holistic optimality principles.

Why is it hard to describe integrated complex systems by means of an overall optimality principles? Integrated robotic systems combine many modules, rely on decentralized but coupled and concurrent computation, control and decision making on various levels of representation and abstraction. It is fair to think of such systems as an aggregation of distributed sub-systems, just like multi-agent systems.

Describing good behavior of distributed agents in terms of a single *central* objective may be intricate in multiple ways. First, deriving optimal agents from central objective criteria (e.g. in the general framework of DEC-POMDPs[3]) is NEXP-complete (Goldman and Zilberstein, 2004). Further, while a central objective might describe the system as a whole concisely, the resulting (optimal) agent might be exceedingly difficult to compute. The distributed agent scenario is an interesting case, where we would like to describe and design each individual agent concisely (perhaps in terms of a local objective function)—but also would like to describe the resulting system behavior (perhaps as approximately following another optimality principle). The relation between the local and system behavior becomes the core question—and is the subject of game theory.[4] Current advances on graphical games (Kearns et al., 2001) make interesting progress on this, borrowing methods from probabilistic inference in graphical models.

In conclusion, for distributed agent systems the optimality approach comes with a series of follow-up problems—and the state-of-the-art practical solutions to such

[3] decentralized partially observed Markov decision processes

[4] Translating from microscopic behavior (described in terms of optimality principles) to the "effective" optimality principles that describes the macroscopic behavior is one of the most interesting and fundamental problems in science per se. Renormalization, cooper pairs, superfluidity, and similar macroscopic theories are examples of this endeavor—or of circumventing this endeavor (Laughlin, 2006).

systems (network routing, swarm behaviors) are typically remote from optimality principles.

Integrated robotic soft- and hardware systems are in many ways similar to distributed agent systems, which discriminates robotics from the areas of machine learning and computer vision. In view of the above discussion, it is therefore a fascinating question whether future robotics research will develop holistic optimality approaches to describe systems. To facilitate this discussion, we consider in the next section alternative design principles.

5 Alternative principles of system design

State-of-the-art robotic systems often involve a number of "principles", represented by diverse concrete approaches such as subsumption architectures, dynamic movement primitives, or hierarchical state machines. It it interesting to discuss to what extent these principles are incompatible or complementary with the optimization stance and point towards alternative approaches to represent systems. Some of the core principles are the following.

Modularity & hierarchy: Modularity is one of the core principles of designing complex integrated systems in software and classical engineering. In robotics, hierarchical finite state machines to organize higher-level behavior, or hierarchically and concurrently combining control attractors have been successful modular design approaches. Further, robotic software systems crucially rely on the current paradigms of software design, including realizations of modularity. Successfully marrying such principles with a holistic optimality approach to systems is hard, as we discussed in the previous section. A possible conclusion is to put more research effort in our understanding how also modularly designed systems can well be represented by optimality principles—or seek for other approaches as sketched in the next section.

One of the pitfalls of modular and hierarchical system representations is the multitude of possible perspectives. We are too easily fooled into thinking we are doing everything correctly when our systems are modular and hierarchical. But, of course, modularity and hierarchy are general and problem-agnostic concepts. The key is to use modularity and hierarchy in a way that reflects the structure of the problem we want to solve. This is again consistent with our view that system representations must reflect problem structure. The key challenge thus must be to uncover this structure and to appropriately reflect it in the hierarchical decomposition of robotic systems. Historically, this decomposition has been into vision, control, planning, reasoning, etc. It is conceivable that a perceived lack of progress in robotics is a consequence of choosing a decomposition that lacks correspondence to the structure of the problems roboticists would like to solve.

Stability & convergence: Dynamic movement primitives, control attractors, control theory as a whole and other approaches, esp. on the motor control level, typically adhere to the principle that the involved processes should guarantee stability and convergence. Many other interesting (computational) processes also guarantee convergence: the relaxation process of physical systems to low (approx. Bethe) energy states (and inference in graphical models), Hebbian adaptation, other learning rules (e.g. gradient descent methods). Optimization processes are also convergent and therefore at least a candidate for the representation of stable and convergent computational processes. In fact, understanding the convergence of a process via a Lyapunov function directly casts the process as an optimization process. So we think that the principles of stability and convergence go well with the optimization stance.

Exploiting algorithmic computational power: The mapping from the objective function to the solution is highly non-linear. Optimization processes therefore leverage substantial computational power in the way they represent systems. This is similarly true for other process representations, such as recurrent neural networks or other complex recurrent system representations. However, while it seems obvious that algorithmic computational power is helpful in designing systems, explicit computation may not be the only way to accomplish this.

Exploiting morphological computational power: Compliant hardware has proven to lead to advanced capabilities in robotic manipulation. Whenever the task is to get into contact with the environment, explicit modeling and model-driven optimal design is problematic. Instead, it seems that the inherent physical interaction processes realized by morphologically clever designs "compute" much better solutions. In these sys-

tems, the optimization problem is solved in a distributed fashion by the hardware itself. The task-specific optimality criteria are encoded directly in the morphology of embodiment. The interesting question is, of course, how such clever morphological designs can be obtained? Even though such designs seem to be currently out of reach for approaches based on optimality, first research efforts are underway to attempt exactly this: to develop optimization methods for task-driven morphological computation.

Taking to heart the combinatorial explosion: Many problems in robotics have been proven to be too computationally difficult to solve exactly in a general setting. This begins with motion planning for polygonal robots among polygonal obstacles in the plane and reaches all the way to inference in POMDPs. Taking this fact seriously must have consequences on how we build systems. Algorithmic concepts, such as completeness, provability of convergence or correctness, etc., must loose their importance. In cases where it is provably impossible to devise general and provably correct algorithms, one has to be content with algorithms that work well in the circumstances encountered most often. Furthermore, all non-trivial proofs pertaining to interactions with the real world make the unrealistic assumption of having a perfect world model. We must realize that an algorithm with provable performance under unrealistic assumptions may perform much worse than a practical but unproven algorithm.

6 Optimization functions vs. optimization processes vs. processes in general

To make an optimization function useful (beyond its communicating value) requires a computational *process* to find the configuration that it implicitly specifies. It is an attractive feature that there exist very general processes (e.g., hill climbing) that permit to locate extremal points for very many different optimization functions. This may have pushed the role of the optimization processes to the backstage ("they do not matter so much...") and the specification of a proper optimization function into the limelight. However, there exists also a large body of research on how *the optimization processes themselves* can be made computationally more efficient. This becomes increasingly inevitable for complex and high dimensional optimization problems—which is the predominantly interesting case for robotics. However, for significant efficiency gains the general optimization algorithm usually has to be replaced by algorithms that are specialized for the type of problem under consideration, which usually requires significant ingenuity and can guarantee the efficiency gains only for the problems within the considered class. For instance, general solutions algorithms for linear optimization could require exponential computation time on "unfavorable" problem instances, until the first algorithm with polynomial scaling was discovered in 1979 (see, e.g. (Traub and Woźniakowski, 1982)) (and this paved the way for the now widely used interior point methods). Thus, for most optimization tasks we end up with associating an optimization "function" with one or several specialized "optimizing processes" that are efficient.

We still may argue: the optimization function "sits in the driver's seat", the process has to follow! But lets look a bit closer at the roles of the two parts: the optimization function specifies a *solution*, the process specifies a *solution path*.[5] For the case of a robot, often the solution path is the desired "product": for instance, when the robot is rejecting a disturbance to restore an optimal configuration, we need in the first place *the process for that movement*, and the underlying cost function is important to the extent that it may have enabled us to find a suitable process. But if we can find a suitable process (or a good approximation) without knowing the cost function, e.g., by some heuristics, the outcome can be perfectly fine for the operation of a robot!

At first sight, abandonment of a cost-function-driven process generation may look less principled than starting from a cost function. However, we all know that for many movements it is rather painful to come up with a cost function from which the movement can be derived.

Self-organizing maps provide a second example: while the original algorithm in the form proposed by (Kohonen, 1982) was derived from heuristic considerations and later shown to be not representable as the gradient dynamics of a cost function, its great simplicity and efficiency has enabled it to become a very versatile tool with numerous applications in robotics, data visualization and classification. Some years after its introduc-

[5] Of course, we could define objective functions over a solution path as well; now desribing desirable optimization processes in terms of optimality principles.

tion, Heskes managed to construct a similarly behaving algorithm (Heskes, 1999) that has the virtue of an exact cost function, but at the price of introducing additional computational complexity in the specification of the map formation process, thereby losing the algorithmic simplicity and computational efficiency of the original, heuristic approach.

Multi-agent systems provide numerous further examples, where extremely simple rules can give rise to exceedingly rich and complex behavior that appears as hard or impossible to represent as the minimization of a simple cost function. A prominent and well-known example is Conway's "game of life" which has been shown to be computationally universal (Rendell, 2011) and thus can mimic any dynamics, depending entirely on its initial conditions ("the program"). An example closer to robotics are the by now historic Braitenberg Vehicles (Braitenberg, 1986), a multi-agent system of simple behaving robots whose action emerged from their particular embodiment and simple sensor-actor connections.

These examples illustrate that there exist cases where a shift of focus away from cost functions and towards processes may lead to remarkable gains in simplicity and efficiency. From this perspective, it should not come as a surprise that state-of-the art robotics is actually very strong in its direct specification of processes (versus cost functions) to create systems that work:

Dynamical movement primitives: The idea to generate a motion from flexibly adjustable families of dynamical systems (Schaal et al., 2005) has by now become a major element in the standard tool set of modern robotics. While some of the employed dynamical systems may be gradient dynamics of a cost function, there is no need for such a constraint and it is easily abandoned, e.g., to enable the control of periodic movements, as required for walking.

Embodied computation moves part of the computation into "the physics of the body" (Pfeifer and Gomez, 2009). While most often used to replace active control of kinematic or stiffness conditions through the physics of a suitably designed material body, physical materials can offer much richer options: already simple friction makes systems dynamics non-conservative. More sophisticated material properties, such as the hysteresis that occurs when an elastic material bends, introduce effects that are hard or even impossible to derive from simple cost functions, while representations directly in terms of processes remain feasible and appear very natural.

Hierarchical State Machines or the related Petri Nets (Thomas et al., 2013; Murata, 1989) describe the behavior of a system as transitions among the nodes of a suitable graph, triggered through externally or internally generated events. This is essentially a discrete process representation, which may, but need not, arise from the minimization of some underlying cost function.

Behavior based architectures are related to HSMs: their behavior emerges from the interaction of "basis behaviors" of a collection of "modules" or "agents" that are coupled directly or through the environment (Arkin, 1998). Again, their basis behaviors may be reasonably simple to describe, but the emerging collective behavior usually is difficult or impossible to represent as some overarching minimization task.

The cost function vs. process view can also be seen mirrored in the two complementary programming paradigms of imperative vs. functional programming: while imperative programming languages require to code a sequence of "statements" that provide a chain of steps towards a goal, functional programming define the program as a set of nested function calls whose evaluation then provides the desired result. Languages such as Prolog come even closer to the cost function perspective in that they just require to specify a number of constraints that need to be obeyed and then automatically find variable assignments that meet these constraints.

We take all this as strong motivation to view the optimization stance as a particular perspective that deserves (and benefits from) being complemented by a *process perspective*. In this "dual view" optimization function view and process view offer complementary descriptions of what a system does, and depending on the case at hand and the given objectives, either one may be superior to the other.

Below we juxtapose the two perspectives with regard to some major aspects.

Explicitness: *Cost functions* specify succinctly a scalar measure of goal achievement. As a result, they are explicit at representing the level of performance of a system. Moreover, to the extent that their algorithm is human-readable, they offer an explicit account of *how* the performance of a system is specified. However, they encode the solution only implic-

itly. This drawback is somewhat mitigated by the fact that—if computational efficiency is not a prime concern—often rather general and simple methods allow to take the step from the cost function to a solution instance. *Processes* specify succinctly *how* a goal can be approached. As a result, they are explicit at representing the solution path, but at the price of saying little or nothing about the attained quality.

Benchmarking: Availability of a cost function makes systems orderable by performance and solves the otherwise difficult issue of benchmarking. However, a general pair ordering (implementability of a "tournament") does not necessarily admit a linear ordering of a given set of objects ("A beats B, B beats C, C beats A"): this again reflects the absence of a global value function in the general case (while "local" comparisons remain possible). This is not a lack at the conceptual level, but a feature of the world and points to the need of making cost functions context dependent, with the context, e.g., given by an opponent (or a population of robots). Multicriteria optimization allows to implement this by having the context choose the mixture weights of the different criteria.

Interface: Cost functions are attractive through their extremely simple 1-dimensional scalar interface: thereby, they provide a very strong "compilation" of a system's behavior and summarize information. This also allows to easily "hide" the implementation of a cost function in black box, without destroying (but perhaps reducing somewhat) its usefulness. In comparison, processes have an inherently more complex interface: typically, they are about structure (and their perhaps simplest incarnation is a state velocity vector) and require a bit more work to encapsulate, but this has become commonplace in today's software, which offers rich support to implement even complex interfaces and create huge libraries of subroutine modules.

7 Conclusion

We provide pointed conclusions from the authors individually.

7.1 By Marc Toussaint

We should further push optimization principles in robotics and do research to overcome the raised issues!

I belief that the benefit of the optimization approach as seen in machine learning and computer vision is transferrable to robotics as well. But it is harder: we need to achieve a holistic understanding of our systems, which are more complex, but we will equally benefit if we are able to transfer this understanding to holistic optimality principles to represent systems.

Let me give a biased summary of the main statements of this paper:

- Optimization principles should be viewed as a means to represent systems. This is an important insight: it clarifies that the optimization stance is a scientific method, a means to represent our understanding of natural and artificial systems systems! Following the optimization stance does not mean to be obsessed with performance; but perhaps (to relate to Oliver Brock's comment below) to be obsessed with a certain methodology that tries to describe systems very concisely. But maybe this is a positive obsession.
- Whether this representational approach is beneficial depends on whether it allows us to capture the structure of the problem—an obvious truth. In our case this means whether the roboticist can express his or her understanding of the problem in a formal objective function or prior over a well-defined hypothesis space.
  But what else could *understanding* possibly mean? How else can we rigorously express the structure of problems other than in terms of priors, objective functions, and hypotheses spaces? Assuming that in the future we better *understood* how we can leverage the mentioned alternative principles to reflect the structure of problems, will this understanding not yet again be expressed in terms of objective functions or priors, now over dynamic processes or over morphologies or over behaviors? What better mathematical tools are there to express understanding?
- Certain optimality principles are infeasible to solve computationally. But first, this does not corrupt their scientific importance, provided they truly capture our understanding of the problem. And second, we should work harder to approximate solutions or find relaxations of the objective.

– Given our *current* knowledge, the implications of distributed optimization principles and processes to describe multi-agent or modular systems raise hard problems. But this should motivate research to better understand distributed and hierarchical optimization processes, their convergence and stability.

Robotics researchers should try to understand their problems and express this understanding rigorously. In my view, expression in terms of priors, objective functions, and hypotheses spaces seems currently the most promising way to express understanding.

The issues raised in this paper motivate highly interesting future research. I find the questions around optimality principles in distributed and hierarchically organized systems particularly interesting and would hope that advances on such questions will also advance the state-of-the-art in robotics.

7.2 By Helge Ritter

The previous discussion showed us that optimization functions and processes provide two different methodologies for describing a system: both connect given boundary conditions (e.g., starting and goal configuration) with a solution, but emphasize different representational means.

Yet we may ask: can we go beyond this and arrive at a stronger unifying perspective, in which processes and optimization appear as particular representational means that both serve a shared, overarching principle?

We are far from being able to provide a ready worked out proposal for this, but we feel that some elements of an overarching perspective are already within reach.

What the two representations share is *a mechanism for the generation of a solution.* This mechanism is represented in two different ways: as a direct process vs. a to-be-minimized scalar function. In robotics (but also in other domains) both representations need to be "unfolded" to bring the solution into existence: the process requires the iterative/recursive application of a mapping that specifies its dynamics locally in (discrete or continuous) time; the same holds for the optimization approach (with time replaced by some abstract iteration time)[6]

This "unfolding" points to a shared principle: the encoding (implemented differently in the two approaches) of a *generator* to perform the unfolding. Usually, this generator is to be applied iteratively, and it can change as a function of the iteration step or the current state (equivalently, we may think of a set of generators, along with a selector mechanism)[7]

Therefore, the unification may arise when we shift the focus from the particular "generator encoding strategies" to the more overarching question about the *intrinsic structure of the appropriate generators and their combination rules.* This stance has a long and fruitful history in mathematics (with the prime number theory, group theory, fractals or linear algebra as prominent examples) and also in physics (e.g., symmetry groups as "generators" of motions or of elementary particles). And it appears pervasive for many well-established methods in robotics: the generation of the reachable configuration space from a group of infinitesimal motions, eigenvector bases or the kernel trick (these being examples in close connection to an extremal principle) to span feature spaces, basis controllers to coordinate robot degree of freedoms in response to external disturbances (an example of the important situation to consider generators embedded in a closed loop!), echo state networks to generate a reservoir of dynamical features, generative grammars to generate discrete structures (such as Lindenmayer systems for geometrical patterns or the various string grammars to generate languages of varying complexity) and many more.

Such "generator stance" will share with the "optimization stance" and the "process stance" the (easy to obtain!) feature of universality. However, its deeper merit will be the shift of perspective, away from particular representations to overarching aspects: what are minimal sets of generators for a required behavior repertoire? How do different choices compare with regard to robustness and information requirements? When are two sets of generators equivalent? How can generators be combined and organized into modular or hierarchical structures? Which generators commute, and which do not? How can a set of generators be extended or focused? And, connecting back to specific representations, what different representations for an equivalence

[6] it should be noted that for the optimization approach the result is always the end point of the generated trajectory, while for the process approach either the end point or the trajectory itself can be the result.

[7] the "generative models" in statistics focus on the parametric specification of a probability density and do not particularly emphasize any process aspect.

class of generators can be found and what computational differences do they offer?

It appears that these (and more) questions may be helpful for embedding the optimization stance and the process stance in a larger landscape and could have useful orienting potential when delving into the unknown territory beyond optimization and heuristic process creation.

7.3 By Oliver Brock

At any point in time, a scientist (or roboticist) has at his or her disposal a number of facts, theories, insights, methodologies, ideologies, etc. Obviously, none of these—let us call them tools—must be ignored. Only if they are all considered can the scientist be sure to have leveraged all possibilities for making progress. But at the same time, one must be aware of the purpose of available tools. Knowledge of purpose will enable the scientist to make an adequate selection of tools to be brought to bear on a specific problem. In many ways, this knowledge is necessary, as there are too many tools available to apply them all. (Interestingly, we have found an explanation for prevalence of "method-monogamists" among the scientists!)

We now have returned to the mantra of this article: the purpose of the tool must reflect the nature of the problem. Put differently, the system's representation must reflect structure inherent to the problem to be solved. So far, I believe, we (the authors, and maybe also the reader) are all in agreement.

In this article we have speculated on the suitability of optimization principles (and some alternatives) for solving problems in robotics. But our discussion of the suitability of tools is moot in the absence of an understanding of the nature of the problem. How should we judge if optimization principles can have a positive impact on robotics? Yes, they may have had one in machine learning. But do problems in robotics and machine learning share fundamental traits? Probably so, but which ones? And which subcategories of problems in robotics share them, as we certainly would not consider it wise to approach all problems in robotics the same way? Of course, we could just try them out. Many people are doing this already. But is this appropriate scientific behavior, or simply a motion through methodological fashions?

So then here is my main criticism: We are obsessed with methodologies, philosophies, and overarching world views. This is just *one* of the two sides we want to match. To perform an appropriate matching between representation and problem we must turn our attention towards the problem! We must try to understand the problem's nature and its structure, chart the different territories spanned by our discipline, determine a topography and taxonomy of problems in robotics. Only then will we be able to understand which tools are best suited for which problems. Must we not admit that we have a much deeper understanding of the principles of, say, optimization than of the principles that govern the behavior of a robotic system? The latter was defined by us above as "an embodied entity that couples internal computational processes with external processes to generate task-directed behavior". The most pressing problem in robotics is that we want to generate *behavior* by coupling internal and external processes. It is not so much our lack of understanding of different methodologies.

My proposal then is to view robotics as an empirical, synthetic science of behavior, similar to what Paul Cohen suggested in the introduction to his book on Empirical Methods for Artificial Intelligence (Cohen, 1995). This endeavor will reveal the underlying structure of problems pertaining to robotic systems and their task-directed behavior when interacting with the world. And then it will become easy to identify which methods we should bring to bear and where to invest further energy in overcoming the many problems we will face, irrespective if we choose optimization principles as a tool or some other methodology. First, we must overcome our obsession with methods, which, I believe, only is a disguise for the fear of truly difficult problems: the problems of the real world, as opposed to the limited worlds covered by any one of the tools we have at our disposal and through whose glasses we have decided to view our world.

References

R. C. Arkin. *Behavior-based robotics.* MIT press, 1998.

V. Braitenberg. *Vehicles: Experiments in synthetic psychology.* MIT press, 1986.

P. Cohen. *Empirical Methods for Artificial Intelligence.* MIT press, 1995.

R. P. Feynman, R. B. Leighton, and M. L. Sands. *The Feynman Lectures on Physics*, volume II. Addison Wesley, 1963.

C. V. Goldman and S. Zilberstein. Decentralized control of cooperative systems: Categorization and complexity analysis. *J. Artif. Intell. Res.(JAIR)*, 22:143–174, 2004.

T. Heskes. Energy functions for self-organizing maps. In E. Oja and S. Kaski, editors, *Kohonen maps*, pages 303–316. Elsevier, 1999.

M. Kearns, M. L. Littman, and S. Singh. Graphical models for game theory. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 253–260. Morgan Kaufmann Publishers Inc., 2001.

T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics 43*, 1:59–69, 1982.

M. C. Koval, N. S. Pollard, and S. S. Srinivasa. Pre-and post-contact policy decomposition for planar contact manipulation under uncertainty. *RSS, Berkeley, USA*, 2014.

R. B. Laughlin. *A different universe: Reinventing physics from the bottom down*. Basic Books, 2006.

J.-P. Laumond, N. Mansard, and J. B. Lasserre. Optimization as motion selection principle in robot action. *Communications of the ACM*, 58(5):64–74, 2015.

J. Moore, R. Cory, and R. Tedrake. Robust post-stall perching with a simple fixed-wing glider using LQR-trees. *Bioinspiration & biomimetics*, 9(2):025013, 2014.

I. Mordatch, Z. Popović, and E. Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144. Eurographics Association, 2012.

T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

R. Pfeifer and G. Gomez. Morphological computationconnecting brain, body, and environment. In B. Sendhoff, E. Körner, O. Sporns, H. Ritter, and K. Doya, editors, *Creating Brain-Like Intelligence*, pages 66–83. Springer Berlin Heidelberg, 2009.

M. Posa, C. Cantu, and R. Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33 (1):69–81, Jan. 2014. ISSN 0278-3649, 1741-3176. doi: 10.1177/0278364913506757.

N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa. CHOMP: Gradient optimization techniques for efficient motion planning. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 489–494. IEEE, 2009.

P. Rendell. A universal turing machine in conway's game of life. In *High Performance Computing and Simulation (HPCS), 2011 International Conference on*, pages 764–772. IEEE, 2011.

S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert. Learning movement primitives. In *In Robotics Research. The Eleventh International Symposium*, pages 561–572, 2005.

S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang, and S. Kim. Design principles for highly efficient quadrupeds and implementation on the mit cheetah robot. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3307–3312. IEEE, 2013.

U. Thomas, G. Hirzinger, B. Rumpe, C. Schulze, and A. Wortmann. A new skill based robot programming language using UML/P statecharts. In *In Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 461–466, 2013.

M. Toussaint. Probabilistic inference as a model of planned behavior. *Künstliche Intelligenz*, 3/09:23–29, 2009. ISSN 0933-1875.

M. Toussaint. Newton methods for k-order markov constrained motion problems. *arXiv preprint arXiv:1407.0414*, 2014.

J. F. Traub and H. Woźniakowski. Complexity of linear programming. *Operations Research Letters*, 1(2):59–62, 1982.

J. Weston, F. Ratle, and R. Collobert. Deep learning via semi-supervised embedding. In *Proc. of the 25th Int. Conf. on Machine Learning (ICML 2008)*, 2008.

D. Zarubin, F. T. Pokorny, M. Toussaint, and D. Kragic. Caging complex objects with geodesic balls. In *Proc. of the Int. Conf. on Intelligent Robots and Systems (IROS 2013)*, 2013.

Marc Toussaint is full professor for Machine Learning and Robotics at the University of Stuttgart since 2012. He studied physics and mathematics at the University of Cologne and received his PhD from Ruhr-University Bochum in 2004 before staying with the University of Edinburgh as a post-doc. In 2007 he received an assistant professorship and Emmy Noether research group leadership at TU & FU Berlin. His research focuses on the combination of decision theory and machine learning, motivated by fundamental research questions in robotics. Reoccurring themes in his research are appropriate representations (symbols, temporal abstractions, relational representations) to enable efficient learning, reasoning and manipulation in real world environments, and how to achieve jointly geometric, logic and probabilistic learning and reasoning. He is coordinator of the German research priority programme *Autonomous Learning*.

Oliver Brock is the Alexander von Humboldt Professor of Robotics in the School of Electrical Engineering and Computer Science at the Technische Universitt Berlin in Germany. He received his Diploma in Computer Science in 1993 from the Technische Universitt Berlin and his Master's and Ph.D. in Computer Science from Stanford University in 1994 and 2000, respectively. He also held post-doctoral positions at Rice University and Stanford University. Starting in 2002, he was an Assistant Professor and Associate Professor in the Department of Computer Science at the University of Massachusetts Amherst, before to moving back to the Technische Universitt Berlin in 2009. The research of Brock's lab, the Robotics and Biology Laboratory, focuses on autonomous mobile manipulation, interactive perception, grasping, manipulation, soft hands, interactive learning, motion generation, and the application of algorithms and concepts from robotics to computational problems in structural molecular biology.

Helge Ritter is professor for neuroinformatics at the Faculty of Technology at Bielefeld University. He studied physics and mathematics at the Universities of Bayreuth, Heidelberg and Munich. After a Ph.D. in physics at Technical University of Munich he stayed at the Laboratory of Computer Science at Helsinki University of Technology and at the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign. In 1990 he obtained a tenured professorship at Bielefeld University. Helge Ritter's main interests are principles of neural computation, cognitive robotics, and interactive intelligent systems. In 1999, Helge Ritter was awarded the SEL Alcatel Research Prize and in 2001 the Leibniz Prize of the German Research Foundation DFG. He is co-founder and Director of the Bielefeld Cognitive Robotics Laboratory (CoR-Lab) and coordinator of the Bielefeld Excellence Cluster Cognitive Interaction Technology (CITEC).