Large Motion Libraries: Toward a "Google" for Robot Motions

Kris Hauser, School of Informatics and Computing, Indiana University at Bloomington
hauserk@indiana.edu

***Abstract*— There is a growing need in robotics for real-time optimal planning and control, driven by the advent new technologies like autonomous vehicles, legged robot locomotion, object manipulation, CAD/CAM, computer animation, and surgical robots. But even at the current state-of-the-art, global optimization is generally too computationally expensive for real-time use. The status quo appears unsuitable looking ahead to the future, which will require addressing progressively higher dimensional systems, faster response rates, longer time horizons, large and detailed environments, and problems with uncertainty. I propose that a motion library approach has the potential to address these upcoming needs. The idea is to first precompute a large library of motion primitives on a set of representative training environments. The robot will then retrieve primitives online to solve novel problems. Given enough training data and perfect recall, performance is limited only by the retrieval cost. The major challenge to address is scale: how many primitives are needed to generalize across all environments and tasks of interest, and how can tools for precomputation and retrieval scale up to thousands or millions of primitives? In this paper, I present a preliminary roadmap for motion library research that will help move toward a "Google" for robot motions.**

Keywords: robotics; optimization; motion planning; machine learning; information retrieval

I. INTRODUCTION

For decades robotics has had to cope with the fact that global optimization is painfully slow, even though although local optimization and solving for feasible suboptimal solutions are generally fast. Moore's law can no longer be relied upon to deliver better performance; although memory cost and storage density continues to improve, CPU speeds and energy costs are starting to plateau (arguably serial performance has already plateaued). The implications are pervasive. Whether the object being optimized is a path, a trajectory, a feedback control policy, a grasp, a geometric quantity, etc., a great deal of human effort must be invested to engineer the environment or devise good heuristics (e.g., initial guesses for local optimizers) to calculate high-quality behaviors. As a result, developing intelligent behavior is time-consuming, even in controlled lab settings.

Can robots automatically learn motion strategies and when to use them? The question of "when to use a motion" is a major challenge, because it requires mapping the space of *problems* (i.e., initial conditions, tasks, and environments) to the space of *optimal motions*. In the worst case, this mapping is intractably complex, but it may be the case that the map can be tractably approximated. For example, the empirical distribution of problems might be approximated by a finite sample, and that problem features are statistically highly correlated with

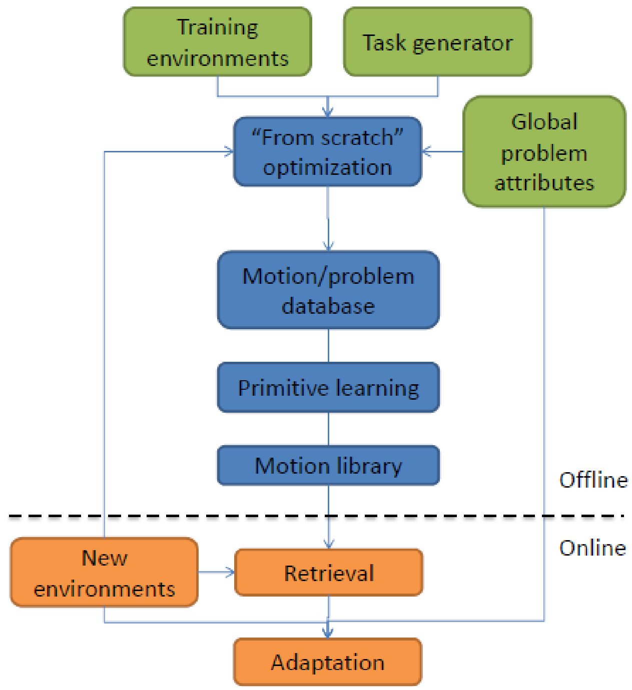


Figure 1. An illustration of the motion library approach.

optimized motions. If the world obeys such a structure, the following motion library approach may be useful (Figure 1):

1. **Precompute** a huge number of problem/optimal motion pairs (motion primitives) and store them in a database (a motion library). Problems will be generated from a set of representative training environments, or from perceptual inputs gathered offline.

2. Online, the robot solves novel problems by **retrieving** appropriate primitives from the library according to a problem similarity metric, and **adapting** them. (More complex forms of adaptation might compose multiple primitives together in sequence through high-level planning, or by blending)

If the motion library were sufficiently rich and retrieval were sufficiently fast, the benefits to such a scheme would be clear: *robots would respond faster*, because extensive optimization on-line would be avoided; *robots would be able to execute unintuitive behaviors at their performance limits*, because optimization is not limited to an engineer's imagination; and *robots would be more capable*, because an essentially infinite number of problem variations can be explored in simulation. Skills will no longer need to be painstakingly-crafted in the lab; an engineer will simply need to provide additional test environments and wait for a modest amount of precomputation time before a new skill emerges automatically.

This paper presents a vision for the new motion library framework and discusses promising research directions for making it feasible. It remains unresolved whether the approach is computationally feasible, whether libraries can be sufficiently rich to cover all problems of interest, and whether retrieval can be made sufficiently fast. But we observe broader computing trends that give us reason to be optimistic. First, library precomputation is trivially parallelizable, and costs are rapidly dropping as vast amounts of computing resources are becoming readily available via high-performance clusters and cloud computing. I argue that with the right computing infrastructure, it would be orders of magnitude cheaper and faster to calculate robot behaviors automatically than to employ human labor to develop them. Second, information retrieval techniques for documents, images, and 3D objects can access relevant queries from databases containing billions of entries in a fraction of a second. Extending them to handle problems and motions will require a great deal of new work, but the challenge is by no means insurmountable.

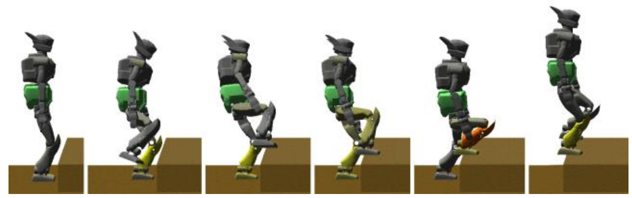
II. Background and Prior Work

The idea of robot learning is appealing, and has been studied in past work in many forms such as reinforcement learning [1], iterative learning control [2], and dynamic movement primitives [3]. However, knowing “when to use” a motion strategy is still a challenge, because rather than learning in the space of states, it requires learning in the space of problems, which is infinite-dimensional. Hence, learning from physical experience or manual teaching typically fails to provide sufficiently large training sets to select appropriate strategies.

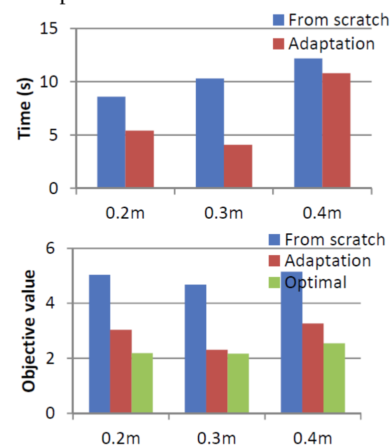
Motion libraries have been studied most significantly in the virtual character animation community. Several techniques exist for generating novel motions from high-quality human motion clips, either by sequencing several motions [4,5] or adapting motions to new characters [6,7]. It is also possible to learn a probability distribution of natural-looking poses from human motion capture data, and to bias the solution of optimization problems toward those poses [8, 9]. The successes of this approach suggest that many complex, multi-step motions can be quickly composed of a relatively small number of simple, reusable subsegments (e.g., stepping motions). This paper outlines a similar approach, but one that does not presuppose the existence of human motion datasets. It also puts a higher priority on physical feasibility through the use of constrained global optimization.

In robotics, past efforts on optimization-based motion learning and adaptation include [10,11,12,13,14]. In general, this research has suggested that a small amount of online optimization to the novel problem (adaptation) makes it less important to learn optimal motions precisely (Figure 2). As a result they are able to use manageably small motion libraries (dozens or hundreds of primitives) to address novel problems. This paper considers the novel research issues that will need to be tackled to scale up to massive numbers of primitives.

(a) Stair step planned entirely from scratch.



(b) Primitive adaptation leads to a more natural looking motion.



(c) Adaptation planning can produce high quality paths in less time than planning from scratch (lower objective values are better).

Figure 2. Adaptation planning for a humanoid robot (reprinted from [14]).

III. Motion Library Approach

This section describes a high level overview of the approach and why it is likely to be economically viable.

A. *The Motion Library Workflow*

Unlike the current state of practice of dedicating tens or hundreds of thousands of man-hours toward engineering robot behaviors, the motion library approach considers the following workflow:

- A robot model, a set of representative training environments, and a task generator are sent to a computing server (e.g., the cloud).
- The server precomputes a massive motion library, including similarity metrics and data structures for optimized retrieval of appropriate motion primitives.
- Either (A) the motion library is transferred to physical robots for local retrieval, or (B) the robots query the motion library remotely from the server.

Periodically, the server may continue to expand the motion library to incrementally improve the robots' performance. Robots may provide feedback about their deployed environments, which helps the motion library adapt over time.

B. Mathematical Formulation

The mathematical formulation of the motion library approach is highly general and straightforward [10]. If p is a problem specification, x is a candidate motion, and $f(x;p)$ is a quality metric (higher is better), we wish to learn an approximation of the map from problems to optimal motions:

$$g(p) \approx x^* \equiv \arg\max_x f(x;p). \tag{1}$$

A motion library is a set of N problem/motion pairs $(p[1],x[1]),\ldots,(p[N],x[N])$. Problems include environmental variables (which are typically complex, e.g., 3D maps) and task variables (e.g., start and goal configuration).To produce the large amounts of training data needed to make learning work, a relatively small number of *manually-provided training environments* is combined with a *task generator*, which samples task variables at random according to a given task distribution.

Given a problem similarity metric d, we can define the primitive retrieval function:

$$retrieve(p) = x[index(p)] \tag{2}$$

where $index(p) = \arg\min_k d(p[k],p)$ is the index of the nearest problem to p. Let us put aside for the moment the issue of defining problem similarity; this issue will be revisited in Section IV.

Now, the final adaptation layer is represented via a map $adapt(x,p)$ from a "guess" x to a solution of problem p. In the simplest unconstrained case, $adapt(x,p)=x$, but more typically the constraints in p will need to be taken into account. The final representation of $g(p)$ is therefore

$$g(p) = adapt(retrieve(p),p). \tag{3}$$

This can be considered a form of nonparametric estimator of the ideal g.

The designer of a motion library must carefully consider the time-quality tradeoff when designing the library size, *adapt* routine, and *retrieve* routine. With respect to solution quality, the method is expected to perform better with N large and strong adaptation, i.e., is likely to map even poor guesses to high quality solutions. With respect to time, this system will perform better when N is low or fast indexing structures help compute (2) quickly, and weak adaptation.

C. The Economic Calculus

Developing planning and control strategies for robots is labor-intensive. Consider that in 2009, Willow Garage developed behaviors for the PR2 household robot to open doors and plug itself in for recharging batteries [15]. A human labor cost from \$50,000-400,000 per robot behavior is estimated given the following assumptions:

- 1-4 engineers at 6-12 months development time per behavior
- \$100,000 annual salary (reasonable for software engineers in Silicon Valley).

On the other hand, with Amazon EC2's cloud computing service, a machine can be rented on-demand for \$0.06/hour [16]. So, \$100,000 buys 1.6 million hours of computation, taking approximately one month of work from 4,000 machines. A motion library approach could compute 70 behaviors for the same cost as an engineer's yearly salary, and at a much faster rate, given the (very rough) assumptions:

- One robust behavior consists of 1,000 optimized motions (e.g., a motion library of size 1,000 successfully solves all problem variations expected to be solved by the behavior)
- A single machine optimizes 1 motion / day.

Granted, these estimates should be taken with a grain of salt. The conversion rate between a "robust behavior" and optimized motion is completely unclear, and the motion optimization speed depends on the complexity of the robot and environment. Furthermore, human labor will still be needed to configure the precomputations by providing test problems, simulations, and problem features, and to verify and test behaviors on the robot. Nevertheless, these calculations suggest that motion libraries have the potential to rapidly accelerate the development of robot behaviors without increasing costs.

IV. New Enabling Technologies

A good motion library will exhibit *fast retrieval* and *wide applicability* while relying only on *weak adaptation*, because online costs are minimized while maintaining high solution quality. Tools from data mining and information retrieval may be useful to learn good libraries from a huge number of raw input motions.

A. Clustering, Problem Features, and Indexing

First, clustering and segmentation may be used to reduce a huge motion space N into a manageable number of motion primitives. Second, due to the complexity of 3D environments, problems will likely need to be indexed by a feature vector rather than a direct representation. It is unclear which of a large number of features may be most predictive of accurate retrievals, but unsupervised feature selection techniques may be useful. Third, approximate nearest neighbors or locality sensitive hashing techniques can be used to achieve sub-linear lookup times even with high N.

B. Adaptation-Sensitive Problem Similarity Metrics

Let us now revisit the issue of problem similarity metrics. Suppose for sake of argument that we could test the adaptation quality for *every* primitive. Then, we would see that the optimal retrieval index $index^*(p)$ is:

$$index^*(p) = \arg\max_k f(adapt(x[k],p),p). \tag{4}$$

Hence, an *optimal* similarity metric d will be one for which $\arg\min_k d(p[k],p)=index^*(p)$ holds over all problems. Such a metric is *adaptation-sensitive* because it depends directly on the adaptation process. Of course, we cannot determine $index^*(p)$ without applying *adapt* to all primitives, which would largely defeat the purpose of a motion library.

However, we can learn a problem-space metric that *approximates* $d(p[k],p) \propto -f(adapt(x[k],p),p)$. Such a Quality-of Adaptation (QoA) metric would result in a close approximation $index(p) \approx index^*(p)$, and hence, high quality adaptations. It is also a simple matter to consider computational costs in the measure of adaptation quality. To train a QoA metric, we may select a sample of source/target (s,t) pairs from the motion library and compute training examples $d(p[s],p[t]) = -f(adapt(x[s],p[t]),p[t]))$. Any supervised method then can be used to learn the function $d(p,p')$.

V. Conclusion and Vision for Future Work

This paper outlined a vision for generating and using motion libraries of unprecedented scale to solve the real-time global optimization problems that are ubiquitous in robotics. The approach is outlined in a mathematically sound framework, and is argued to be economically viable compared to human labor in generating robust robot behaviors.

Future research should address whether a library can represent repeatable motion patterns:

1. *How large must a motion library be to tackle complex problems*? Existing techniques can handle dozens of primitives, but new retrieval techniques are needed to scale to thousands or millions. If billions or trillions of motions are needed, then the approach is likely impractical.

2. *How can common motifs* (steering maneuvers, footsteps, grasps) *be clustered and segmented* to be used as primitives in the vast amounts of generated data?

And how to implement primitive retrieval:

3. *What problem features and indexing structures yield fast and effective primitive retrieval*? Image retrieval techniques scale to millions of images due to decades of research in high-quality image features (e.g., SIFT, HOG) and approximate nearest neighbors techniques, while research on robot control problem retrieval is practically nonexistent.

4. *How does the power of the adaptation routine affect library applicability and responsiveness*? Stronger adaptation lessens the need for larger libraries, at the expense of more online computation.

This paper outlined a number of promising approaches for addressing these research challenges, and sketched out the idea of learning adaptation-sensitive problem similarity metrics.

Outside of the scope of this paper, but still important for the success of a motion library method, are research directions in closing the loop with high-level planning and perception:

5. How can planners efficiently compose long-term, high-level behavior out of primitives, particularly where multiple primitives appear equally favorable to achieve a goal?

6. How should the robot incorporate sensing feedback to compensate for simulation errors?

Should this effort be successful in overcoming the intractability of global optimization, it could usher in dramatic advances in real-time control of complex tasks, such as those that are typical in household, industrial, and space robots.

References

[1] L.P. Kaelbling, M.L. Littman, A.W. Moore. *Reinforcement learning: A survey*. Journal of Artificial Intelligence Research, Vol 4, pages 237-285, 1996.

[2] K.L. Moore. *Iterative learning control: an expository overview*. Applied and computational control, signals, and circuits, 1999.

[3] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert. *Learning movement primitives*. Robotics Research, Springer Tracts in Advanced Robotics, vol 15, pages 561-572, 2005.

[4] Okan Arikan and David A. Forsyth. *Interactive motion generation from examples*. ACM Transactions on Graphics (ACM SIGGRAPH 2002), 21(3):483-490, 2002.

[5] Lucas Kovar, Michael Gleicher, and Frederic Pighin. *Motion graphs*. In SIGGRAPH, pages 473-482, San Antonio, Texas, 2002.

[6] Michael Gleicher. *Retargetting motion to new characters*. In SIGGRAPH, pages 33-42, 1998.

[7] Zoran Popovic and Andrew Witkin. Physically based motion transformation. In SIGGRAPH, pages 11-20, 1999.

[8] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popovic. *Style-based inverse kinematics*. ACM Trans. Graph., 23(3):522-531, 2004.

[9] Katsu Yamane, James J. Kuffner, and Jessica K. Hodgins. *Synthesizing animations of human manipulation tasks*. ACM Trans. Graph., 23(3):532{539, 2004.

[10] Nikolay Jetchev and Marc Toussaint. *Trajectory prediction: learning to map situations to robot trajectories*. Proceedings of the 26th International Conference on Machine Learning (ICML-09), 2009.

[11] Nikolay Jetchev and Marc Toussaint. *Fast motion planning from experience: trajectory prediction for speeding up movement generation*. Autonomous Robots. 34(1-2): 111-127, January 2013.

[12] A. Dragan, G. Gordon, and S. Srinivasa. *Learning from Experience in Manipulation Planning: Setting the Right Goals*. Int'l Symposium on Robotics Research, 2011.

[13] Berenson, D.; Abbeel, P.; Goldberg, K. *A robot path planning framework that learns from experience*. IEEE Int'l Conference on Robotics and Automation (ICRA), May 2012.

[14] Kris Hauser, Timothy Bretl, Kensuke Harada, and Jean-Claude Latombe. *Using motion primitives in probabilistic sample-based planning for humanoid robots*. In WAFR, New York, NY, 2006.

[15] J. Markoff. *Opening Doors on the Way to a Personal Robot*. New York Times, June 8, 2009.

[16] Amazon EC2 Pricing. http://aws.amazon.com/ec2/pricing/ (retrieved on 5/1/2013).