In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, California, 2011

Planning Humanlike Actions in Blending Spaces

Yazhou Huang, Mentar Mahmudi and Marcelo Kallmann

***Abstract*— We introduce an approach for enabling sampling-based planners to compute motions with humanlike appearance. The proposed method is based on a space of blendable example motions collected by motion capture. This space is explored by a sampling-based planner that is able to produce motions around obstacles while keeping solutions similar to the original examples. The results therefore largely maintain the humanlike characteristics observed in the example motions. The method is applied to generic upper-body actions and is complemented by a locomotion planner that searches for suitable body placements for executing upper-body actions successfully. As a result, our overall multi-modal planning method is able to automatically coordinate whole-body motions for action execution among obstacles, and the produced motions remain similar to example motions given as input to the system.**

I. Introduction

Despite several successes in the motion planning domain, achieving whole-body humanoid motions with humanlike characteristics remains a challenge. One main difficulty that emerges from this problem is to strike the right balance between how much to explore solutions during motion planning and how much to constrain the search space in order to obtain solutions with humanlike characteristics.

Our proposed method starts by decomposing the problem into a multi-modal planning problem, where basic skills are planned individually and also coordinated with each other. This approach facilitates addressing the specific needs of each skill, and is inspired by how humans may solve real motion planning problems. This paper addresses the case of planning motions composed of two skills, locomotion and generic upper-body actions, and we use example motions from motion capture as a way to build search spaces with humanlike characteristics.

For example, consider the simple scenario where a character walks toward a light switch and turns it on, as shown in Figure 1. Solving this problem requires a locomotion planner able to place the character in a suitable location near the switch, and then an upper-body action is required for reaching and pushing the light switch. Our overall method solves such class of full-body motion problems.

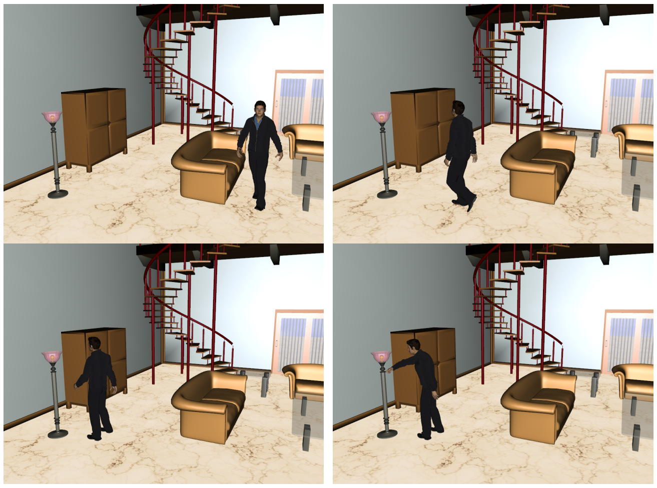

Fig. 1. Our overall planning approach achieves precise end-effector placement for action execution among obstacles and produces humanlike results in coordination with locomotion.

Upper-body actions are synthesized with a novel motion planner that searches for solutions in a space of blendings between example motions. The method produces collision-free motions to precise targets and achieves solutions with humanlike characteristics similar to the ones observed in the original example motions. The planner is therefore limited to explore the variations embedded in the example motions, and a suitable body placement is essential for enabling the upper-body planner to be successful.

The upper-body planner is thus complemented with a standard locomotion planner based on a motion graph. The locomotion planner will search for suitable body placements in coordination with the upper-body planner until an overall whole-body motion for performing the target upper-body action is found.

As a result, our combined approach is able to automatically coordinate locomotion with generic actions, and the produced motions are realistic, collision-free, and can precisely interact with the environment. Our overall method is able to address a broad range of real-life tasks and is therefore useful to a number of applications in ergonomics, training, education, entertainment, and also humanoid robotics.

II. Related Work

Traditional motion planning methods [24, 27, 29] are based on the systematic search of configuration spaces. Among the several techniques, sampling-based methods [18, 23, 28] have become extremely popular for planning in continuous configuration spaces. Such methods are also popular for planning humanoid motions, in particular for planning footsteps [4, 5, 21] and reaching motions [2, 7, 8, 15, 19].

The authors are with the School of Engineering of the University of California, Merced, N. Lake Road, Merced CA 95343. {yhuang6,mmahmudi,mkallmann}@ucmerced.edu

Multi-modal planning has recently emerged for humanoids and has been developed for locomotion and climbing in difficult terrains [3, 11, 12], and also to sequentially coordinate walking and pushing [13]. With a focus on locomotion, extensions to the basic PRM method for handling multi-modal problems have been proposed [10], and generic multi-skill planners have been developed [17]. However, no previous work in motion planning has addressed the computation of motions with humanlike characteristics.

In contrast, methods originated from the computer animation area focus on achieving humanlike results from motion capture, without much importance given to searching for collision-free motions in complex environments. Probably the most popular approach for computing realistic full-body motions is to extract motions from a motion graph structure. Motion graphs are built by connecting the frames of high similarity in a database of motion capture examples [1, 20, 30, 31, 34, 35]. Once the motion graph is available, a graph search is performed in order to extract motions with desired properties. The main drawback of motion graphs is that a prohibitively large structure would be needed in order to produce motions satisfying many constraints, such as around obstacles and addressing precise placements of end-effectors.

Planning methods have been integrated with motion capture data in many ways. For instance, Lau and Kuffner [25] plan over a behavior-based finite state machine of motions [26], Choi et al [6] combine motion capture with probabilistic roadmaps, and many other planning methods have been proposed for synthesizing full-body motions among obstacles [9, 16, 22, 26, 33]. However, none of these methods have proposed a solution for planning generic upper-body actions in a continuous space and in coordination with locomotion. The ability to search in a continuous action space allows planners to compute much more complex solutions. Our proposed method represents the first approach for solving this problem with humanlike results, and is based on a novel sampling-based search defined on a space of motion blendings.

III. Locomotion Planner

We start from the observation that correct body positioning is essential for the execution of humanlike upper-body actions. A locomotion planner is therefore needed to explore suitable body placements nearby the action target location. Any locomotion planner could be integrated in our overall approach to the problem, and in this paper we have developed a locomotion planner based on a motion graph.

We construct our locomotion graph similarly to Kovar et al [20], however using a more efficient segmentation procedure designed for detecting useful walking cycles. The procedure is based on feature-based segmentation rules, similarly to the ones employed by Müller et al [32]. Each frame in the motion capture database is tested against a foot-crossing binary test which looks whether the right ankle of the character is behind or in front of the plane created by the left hip, right hip and the left ankle joint positions. All frames in a motion capture sequence that lead to a change in the binary test become candidate transition frames for the final locomotion graph.

After all motion clips are segmented, a pairwise test between the candidate transitions is then performed only between the initial and final frames of each segmented clip in order to determine the acceptable transitions. We use the same distance metric alignment transformation as in the original motion graph work [20]. With our segmentation method the graph construction is sped up dramatically in comparison to the traditional pairwise comparison among all pairs of frames. For instance in a database with 567 motions, the graph construction time improved from 2min to 1s, and for a database with 2245 motions, the improvement was from 32min to 11.5s. No drawbacks were detected for employing our simplified segmentation procedure and the locomotion planning results remained of high quality.

Let q_i represent the initial full-body posture of the character. The task of the overall planning problem is to find a full-body motion composed of two parts: locomotion for body positioning, and then upper-body action execution satisfying a given end-effector goal location p_g. The goal location may be a position target to point to, a 6 degrees of freedom vector encoding position and orientation of a precise hand placement for grasping, etc. The set Q_g denotes all possible body postures satisfying the action goal point p_g.

The task of the locomotion planner is to explore suitable body placements for enabling the action planner to reach a posture in Q_g. Once the locomotion graph is available, an A* search for *unrolling the graph* is employed with the cost-to-go heuristic attracting the search towards p_g, and only allowing collision-free motions to be expanded. Figure 2 illustrates several expansions obtained with the unrolling of the motion graph.

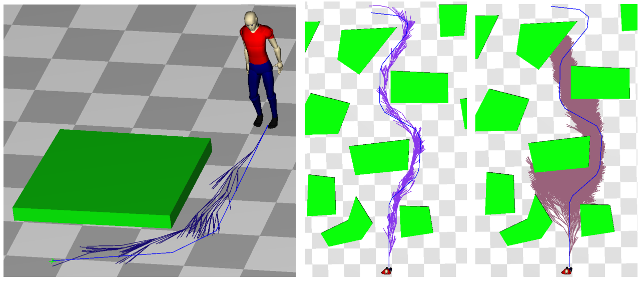

Fig. 2. The images illustrate expansions obtained by unrolling the locomotion graph. The expansions generate several possible trajectories nearby obstacles (left), which will be later considered for upper-body action execution. If a long locomotion is expected, the branching factor of the search can be constrained to only allow expansions nearby a 2D path planned on the floor plan (center), otherwise too many unnecessary expansions may occur (right). A large literature on motion graphs is available with several extensions that can be integrated to customize the locomotion planner to specific situations.

Whenever the locomotion search expansion generates a character posture q_a that is close enough to p_g, q_a is then considered as a transition point to the upper body action and q_a becomes the initial posture for the upper-body action planner, which will in turn launch a bidirectional search attempting to reach a posture in Q_g. The upper-body planner is explained in Section IV.

If the upper-body planner is not successful after a fixed number of iterations, the locomotion planner continues to expand towards additional candidate body placements until the action can be executed or until a maximum time limit is reached, in which case the overall planner returns failure. Figure 3 illustrates the bi-modal overall search procedure.

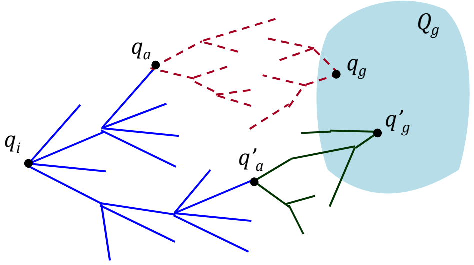

Fig. 3. The locomotion planner unrolls motion capture locomotion clips (left blue tree), and for each candidate initial action posture, a bidirectional upper-body action planning is performed. Here q_a was not able to generate a solution after a fixed maximum number of iterations and a new body placement q'_a was finally successful.

IV. Upper-Body Action Planner

Every time one branch of the locomotion graph expansion reaches a character pose q_a that is close enough to the action target, q_a becomes a candidate initial pose for initializing the action planner. Selecting a suitable proximity threshold is action-dependent. For example, for reaching motions a suitable distance will be related to the length of the character's arm, while for pointing actions larger values can be used for enabling pointing to targets from a certain distance.

The upper-body action is specified with a database of similar and time-aligned example motions, which are realistic upper-body action instances collected from motion capture.

Let a motion $\mathbf{M}(t)$ be represented as a sequence of poses with a discrete time (or frame) parameterization t. With the character at pose q_a, we employ an Inverse Blending optimization procedure [14] in order to obtain a set of blending weights $\mathbf{w_g}$ creating an upper-body action motion $\mathbf{M}(\mathbf{w_g})$ precisely reaching the action target p_g:

$$\mathbf{M}(\mathbf{w_g}) = \sum_{j=1}^{k} w_j \mathbf{M_j}, \qquad \mathbf{w_g} = \{w_1, \ldots, w_k\}, \qquad (1)$$

where k is a fixed parameter specifying the number of example motions from the database to be considered for blending. For a given p_g the k example actions that reach

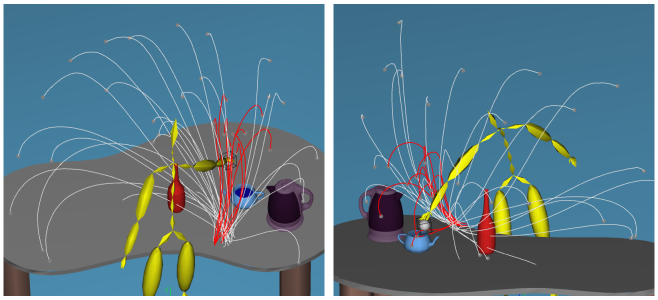

Fig. 4. The images show the trajectories of the character's hand from example motions in a database of 36 pointings. The character's pose shows the final posture of the motion obtained by blending the 10 examples with the trajectories in red ($k = 10$). The blending weights were obtained by inverse blending optimization. The blended pointing motion successfully reaches the action target p_g, which is represented by the gray sphere.

locations closest to p_g are selected. Figure 4 illustrates one database of example motions used in our experiments.

In our experiments $\mathbf{w_g}$ can be determined by inverse blending optimization under 2 milliseconds of computation. The routine returns when the error (proximity to target p_g) reaches zero or cannot be further minimized, in which case failure is returned and q_a is discarded as a candidate body placement.

Whenever $\mathbf{w_g}$ is successfully computed from the inverse blending procedure, motion $\mathbf{M}(\mathbf{w_g})$ provides a realistic humanlike upper-body action precisely meeting the action goal p_g. Motion $\mathbf{M}(\mathbf{w_g})$ is then tested with several discrete collision checks over its time parameterization interval $[t_a, t_g]$. If no collisions are detected the overall planning problem is solved.

The planner: when motion $\mathbf{M}(\mathbf{w_g})$ collides with the environment, all the non-valid frames are removed and the motion is split in two pieces: the first piece contains the adjacent valid frames starting from time t_a, and the second piece contains the adjacent valid frames leading to the frame at time t_g. These motion pieces will define two initial trees T_1 and T_2 to initialize the bidirectional action planner. A few frames equally spaced in time are taken from each motion piece to define the initial nodes of the trees. The upper-body planner is only initialized if posture q_g at time t_g is a valid posture, therefore trees T_1 and T_2 can be always initialized since posture q_a at time t_a is a valid posture determined by the locomotion planner.

With the initial search trees defined a bidirectional search procedure is repetitively called until a collision-free motion can be found to reconnect the two trees. The state space of the search is a weight-temporal space. The nodes stored in trees T_1 and T_2 will each contain a pair $(\mathbf{w}, t)$ where the blending weight vector $\mathbf{w}$ specifies the posture (by direct blending) associated with the node, and the time t indicates at which time that posture is specified along the motion being planned. In other words, each node $(\mathbf{w}, t)$ specifies a posture $q(\mathbf{w}, t)$ that is the frame of $\mathbf{M}(\mathbf{w})$ at time t.

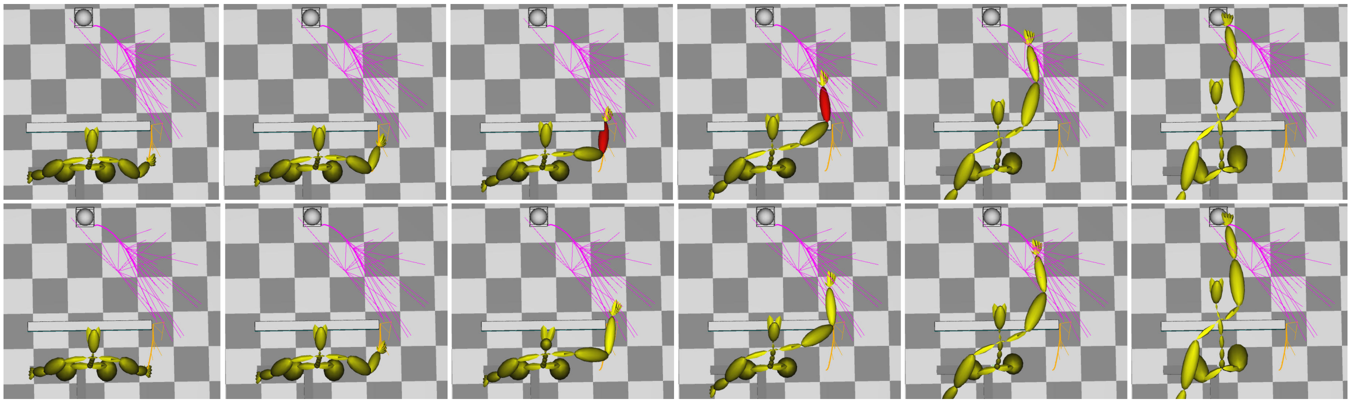

Fig. 5. The top sequence illustrates the invalid frames in $\mathbf{M}(\mathbf{w_g})$ due to collisions, and the bottom image illustrates the solution motion obtained by the action planner. The images also show the expanded nodes of trees T_1 and T_2. The thicker edges represent the initial branches that were used to initialize the trees. The search is performed in the weight-time space and the position of the wrist joint was used to plot the edges of the search trees.

The strategy of searching in the weight-temporal space is the key element of the proposed planner: it enables a search procedure that explores the motion variations available in the set of example action motions. The planner will never synthesize new motions from scratch but instead will search for combinations of existing motion variations. The solution plan is a sequence of time parameterized blending weights that are afterwards interpolated to achieve a smooth solution motion with varying contributions from the example actions. See figure 5 for an example.

The bidirectional search routine performs successive search expansions until trees T_1 and T_2 are connected, or until a maximum of N_{max} iterations pass, in which case failure is returned. The expansion routine of the action planner is detailed in Algorithm 1. At each expansion the algorithm tries to connect the trees, and if not possible it tries to grow each tree in the direction of the random time sample t_{rand} by a time interval $t_\triangle$. The details of the algorithm are explained in the paragraphs below.

Algorithm 1 Action Planner Expansion Routine

```
Expand_Bidirectional_Search ( $p_a$, $\mathbf{w}_g$ )
 1. $T_1 \leftarrow$ tree starting at $q_a$
 2. $T_2 \leftarrow$ tree reaching $q_g$
 3. $(\mathbf{w}_{rand}, t_{rand}) \leftarrow$ sample in weight-time space
 4. $(\mathbf{w}_1, t_1) \leftarrow T_1$ node closest to $t_{rand}$, $t_1 < t_{rand}$
 5. $(\mathbf{w}_2, t_2) \leftarrow T_2$ node closest to $t_{rand}$, $t_{rand} < t_2$
 6. if ( connection $(\mathbf{w}_1, t_1)$ to $(\mathbf{w}_2, t_2)$ is valid ) then
 7.     return CONNECTION_FOUND
 8. else
 9.     try_to_expand ( $(\mathbf{w}_1, t_1)$, $(\mathbf{w}_{rand}, t_{rand})$, $t_\triangle$ )
10.     try_to_expand ( $(\mathbf{w}_2, t_2)$, $(\mathbf{w}_{rand}, t_{rand})$, $t_\triangle$ )
11.     return NOT_YET_FOUND
12. end if
```

Sampling the weight-time space: in line 3 of the algorithm a $(\mathbf{w}_{rand}, t_{rand})$ pair is sampled. Time t_{rand} is randomly sampled in interval $[t_a + t^a_\triangle, t_g - t^b_\triangle]$. The $t^{a,b}_\triangle$ parameters concentrate the sampling on the parts of the environment where collisions are most often found, since the extremities of the original motion $\mathbf{M}(\mathbf{w_g})$ are always valid.

A more specific strategy for sampling the weight space is required for the algorithm to well explore the motion variations embedded in the example actions being blended. We first sample one example motion index m_{rand} among the full set of example motions in the database. We also bias the index sampling to select more samples outside of the k motions used by the initial inverse blending solution in order to favor variations and avoid duplication of blended motions. Then a relatively large blending weight w_{rand} is sampled (in interval $[0.5, 1.0]$) and associated to the motion with index m_{rand}. The new weight w_{rand} is incorporated to the overall weight vector $\mathbf{w}_{rand}$ and the other weights are lowered by uniform scaling such that $\mathbf{w}_{rand}$ is re-normalized to 1 and the influence from example motion m_{rand} remains higher than the other influences. Our experiments have showed that this procedure generates more collision-free samples and yields a better overall planning performance than a simplistic uniform sampling over the overall weight space.

Node expansion: after a node $(\mathbf{w}_{rand}, t_{rand})$ is sampled, the algorithm will then select on each tree the nodes $(\mathbf{w}_1, t_1)$ and $(\mathbf{w}_2, t_2)$ that are closest in time to t_{rand}, respecting the monotone time condition $t_1 < t_{rand} < t_2$, and then attempt to connect the two nodes (line 6 in the algorithm). If a connection is found the algorithm successfully terminates, otherwise the two trees are expanded.

Routine try_to_expand() in lines 9 and 10 of the algorithm will try to grow each tree in the direction of $(\mathbf{w}_{rand}, t_{rand})$ by a time step $t_\triangle$. This procedure yields a new node $(\mathbf{w}'_1, t'_1)$ to be connected to $(\mathbf{w}_1, t_1)$ on T_1. Weight vector $\mathbf{w}'_1$ is an interpolation between $\mathbf{w}_1$ and $\mathbf{w}_{rand}$ at time t'_1, where $t'_1 = t_1 + t_\triangle$, adjusted such that $t'_1 \leq t_{rand}$. A discrete collision test is performed for $q(\mathbf{w}'_1, t'_1)$ and it is added to T_1 only if no collisions are detected. An equivalent expansion procedure is performed to add node $(\mathbf{w}'_2, t'_2)$ to tree T_2.

Lazy collision tests: the algorithm postpones fine resolution

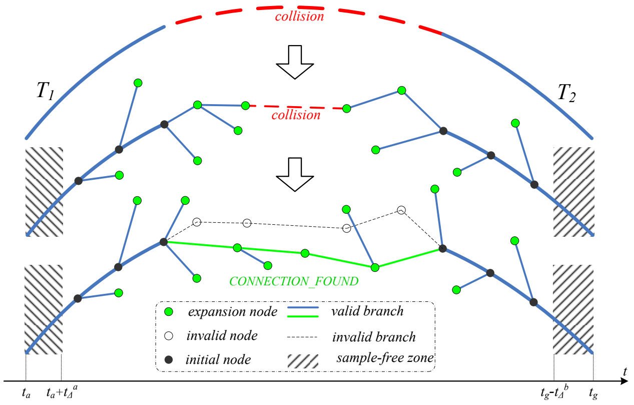

Fig. 6. Illustration of the bidirectional expansion procedure.

of collision tests along tree edges until the trees are connected, in order to promote fast exploration of the search space. When the trees connect, discrete collision tests are performed over the segments composing the solution path that have not yet been tested for collision, and a solution is found if no collisions are detected. If collisions are found, the connection between the trees is broken at the invalid edge and the expansion iterations re-start in search of new connections. When a solution is found, the solution motion is generated by smoothly interpolating (with ease-in and ease-out interpolators) each pair of motions generated by adjacent weight vectors in the solution plan. Such interpolated concatenation of motion pieces is illustrated in Figure 7.

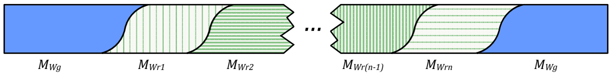

Fig. 7. Nodes $(\mathbf{w}, t)$ along a tree branch are concatenated with smooth transitions to form the corresponding motion.

Extensions: We have also obtained improvements with the popular strategy of sampling nearby obstacles. We identify the nodes that, after a few iterations, could not connect to samples due collisions. These nodes indicate proximity to obstacles. The algorithm can then prioritize the sampling (temporally) near these nodes, within a window of $\pm t_\Delta$, so that the success rate of finding collision-free nodes around these locations is improved.

Several other extensions have been integrated in our framework, in particular, the gaze of the character can be independently controlled to fixate the action target and the obstacles. The velocity profile of the end-effector in each solution motion obtained is also smoothed as a final post-processing operation in order to maintain the final motion with a velocity profile close to the velocity profile of the most similar motion (in duration) in the database.

V. Results and Discussion

Several animations have been produced for demonstrating the results of our overall method. See Figure 1, 8 and 9 for examples. The accompanying video also presents several obtained results. The locomotion planner quickly expands several body placements nearby the action targets, providing the upper-body planner several options for planning the upper-body action. The overall full-body planner therefore evaluates different body placements until a suitable valid and collision-free action is found.

The decoupling between locomotion and upper-body actions avoids several problems related to large motion graphs including both locomotion and upper-body actions. The weight-time space of action blendings provides a continuous space for planning where only the humanlike strategies encoded in the example actions are explored. The action planner uses a relatively large t_Δ in order to quickly explore the space around the obstacles, such that the action planner can quickly return to the locomotion planner in case of failure, and it also minimizes jerky variations in found solutions. In practice the planner showed to always produce smooth motions with the only post-processing operation being the tuning of the final velocity profile for the end-effector. No other post-processing smoothing procedures were required. The solution motion always resembles the given examples in the action database and "no surprises"' in the final motion will appear.

The performance of the planner largely depends on the considered example motions and on the number of triangles considered for collision detection. In the example in Figure 8 a database of 36 motions (shown in Figure 4) was used and 5K triangles were considered for collision detection. A 10000-trial benchmarking test of the action planner with random obstacle and target locations found solutions in 78% of the cases, with each bidirectional search taking in average 4.92 milliseconds. For the remaining cases the planner returned failure after hitting the limit of 500 expansions within an average of 12.76 milliseconds of computation. These times demonstrate that an extensive overall search composed of both locomotion and action planning can be executed in less than one second in reasonably complex environments.

VI. Conclusions

This paper describes a multi-skill motion planning approach that integrates discrete search in a locomotion graph with a systematic bidirectional expansion in a novel weight-time action search space. The search spaces are based on motion capture data, ensuring the achievement of humanlike results. The approach for the upper-body action planning represents the first sampling-based search algorithm defined on a continuous space based on humanlike motion capture examples. The overall approach is able to automatically coordinate locomotion with generic actions among obstacles, and the produced motions are realistic, collision-free, and precisely meeting given targets in the environment. Since actions are modeled from motion capture examples, any upper body actions can be planned, from reaching motions and pointings to generic gestures.

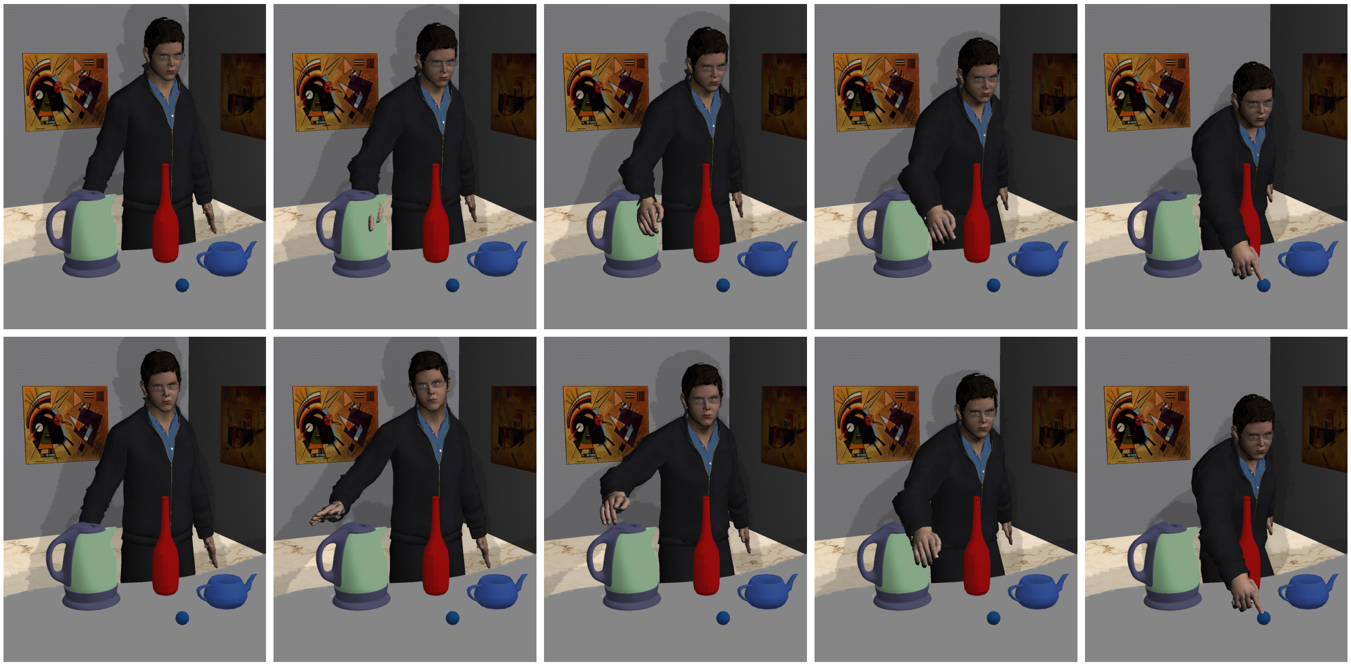

Fig. 8. The top sequence shows the initial motion computed by inverse blending. The initial motion collides with the environment and it is therefore used to initialize the action planner (Algorithm 1). The bottom sequence illustrates the collision-free solution obtained with the planner. Since the planner is restricted to search in a blending space of example actions, the particular strategy obtained in the solution also exists in the database of example actions. The solution of "pointing over obstacles" can therefore be considered to be a humanlike solution.

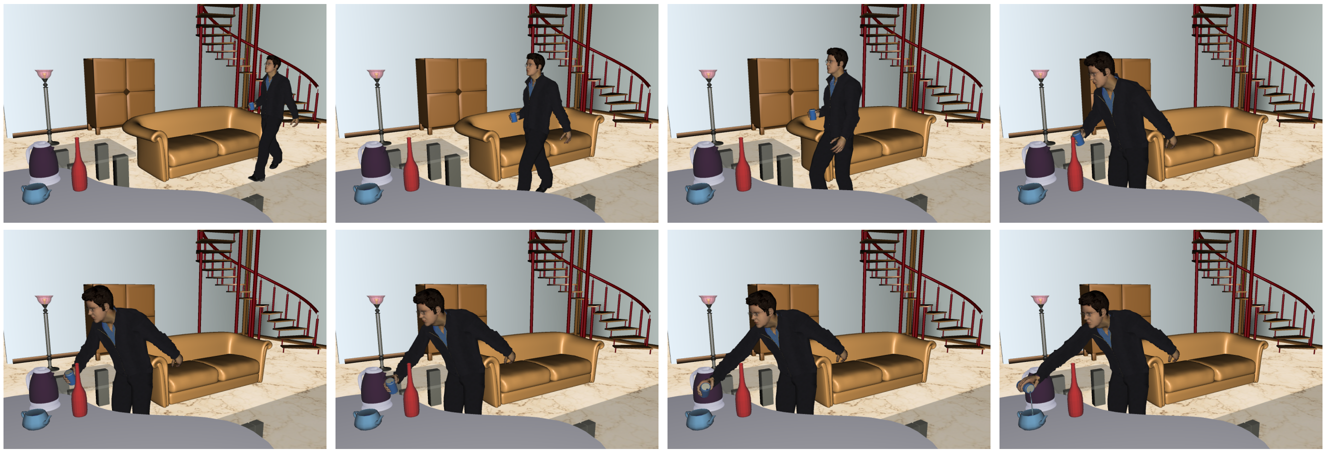

Fig. 9. Example of pouring water on a distant location on a table, among obstacles, and coordinated with locomotion. The behavior of the left arm going backwards in order to assist with balance completely comes from the example motions being blended by the action planner.

Acknowledgments This work was supported by NSF Award IIS-0915665.

References

[1] O. Arikan and D. A. Forsyth. Synthesizing constrained motions from examples. *Proceedings of SIGGRAPH*, 21(3):483–490, 2002.

[2] D. Bertram, J. Kuffner, R. Dillmann, and T. Asfour. An integrated approach to inverse kinematics and path planning for redundant manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1874–1879. IEEE, May 2006.

[3] T. Bretl. Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem. *International Journal of Robotics Research*, 25(4):317–342, 2006. ISSN 0278-3649.

[4] J. Chestnutt, M. Lau, K. M. Cheung, J. Kuffner, J. K. Hodgins, and T. Kanade. Footstep planning for the honda asimo humanoid. In *ICRA*, April 2005.

[5] J. Chestnutt, K. Nishiwaki, J. Kuffner, and S. Kagami. An adaptive action model for legged navigation planning. In *Proceedings of the IEEE/RAS International Conference on Humanoid Robotics*, 2007.

[6] M. G. Choi and J. Lee. Planning biped locomotion using motion capture data and probabilistic roadmaps.

ACM Transactions on Graphics, 22:182–203, 2003.

[7] R. Diankov and J. Kuffner. randomized statistical path planning. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 1–6, May 19-23 2008.

[8] E. Drumwright and V. Ng-Thow-Hing. Toward interactive reaching in static environments for humanoid robots. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, October 2006.

[9] C. Esteves, G. Arechavaleta, J. Pettré, and J.-P. Laumond. Animation planning for virtual characters cooperation. *ACM Transaction on Graphics*, 25(2):319–339, 2006. ISSN 0730-0301.

[10] K. Hauser and J. Latombe. Multi-modal motion planning in non-expansive spaces. In *Proceedings of the 8th Workshop on Algorithmic Foundations of Robotics (WAFR)*, December 7-9 2008.

[11] K. Hauser, T. Bretl, and J. Latombe. Non-gaited humanoid locomotion planning. In *Humanoids*, pages 2641– 2648, December 2005.

[12] K. Hauser, T. Bretl, K. Harada, and J. Latombe. Using motion primitives in probabilistic sample-based planning for humanoid robots. In *Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 2641– 2648, July 2006.

[13] K. K. Hauser, V. Ng-Thowhing, Gonzalez-Baos, H. Mukai, and S. Kuriyama. Multi-modal motion planning for a humanoid robot manipulation task. In *International Symposium on Robotics Research*, 2007.

[14] Y. Huang and M. Kallmann. Motion parameterization with inverse blending. In *Proceedings of the Third International Conference on Motion In Games*, Berlin, 2010. Springer.

[15] S. Kagami, J. Kuffner, K. Nishiwaki, M. Inaba, and H. Inoue. Humanoid arm motion planning using stereo vision and RRT search. *Journal of Robotics and Mechatronics*, April 2003.

[16] M. Kallmann, A. Aubel, T. Abaci, and D. Thalmann. Planning collision-free reaching motions for interactive object manipulation and grasping. *Computer graphics Forum (Proceedings of Eurographics'03)*, 22(3):313–322, September 2003.

[17] M. Kallmann, Y. Huang, and R. Backman. A skill-based motion planning framework for humanoids. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2010.

[18] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars. Probabilistic roadmaps for fast path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12:566–580, 1996.

[19] Y. Koga, K. Kondo, J. J. Kuffner, and J.-C. Latombe. Planning motions with intentions. In *Proceedings of SIGGRAPH*, pages 395–408. ACM Press, 1994.

[20] L. Kovar, M. Gleicher, and F. H. Pighin. Motion graphs. *Proceedings of SIGGRAPH*, 21(3):473–482, 2002.

[21] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue. Motion planning for humanoid robots. In *Proceedings of the 11th International Symposium of Robotics Research (ISRR)*, November 2003.

[22] J. J. Kuffner and J.-C. Latombe. Interactive manipulation planning for animated characters. In *Proceedings of Pacific Graphics*, Hong Kong, October 2000. poster paper.

[23] J. J. Kuffner and S. M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, April 2000.

[24] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publisher, December 1990.

[25] M. Lau and J. J. Kuffner. Behavior planning for character animation. In *2005 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 271–280, Aug. 2005.

[26] M. Lau and J. J. Kuffner. Precomputed search trees: planning for interactive goal-driven animation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA)*, pages 299–308, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[27] J.-P. P. Laumond. *Robot Motion Planning and Control*. Springer-Verlag, NJ, USA, 1998.

[28] S. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Iowa State University, Computer Science Department, October 1998.

[29] S. M. LaValle. *Planning Algorithms*. Cambridge University Press (available on-line), 2006.

[30] J. Lee, J. Chai, P. Reitsma, J. K. Hodgins, and N. Pollard. Interactive control of avatars animated with human motion data. *Proceedings of SIGGRAPH*, 21(3):491–500, July 2002.

[31] Y. Li, T.-S. Wang, and H.-Y. Shum. Motion texture: a two-level statistical model for character motion synthesis. *Proceedings of SIGGRAPH*, 21(3):465–472, 2002.

[32] M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. In *Proceedings of SIGGRAPH*, pages 677–685, New York, NY, USA, 2005. ACM Press.

[33] J. Pan, L. Zhang, M. Lin, and D. Manocha. A hybrid approach for synthesizing human motion in constrained environments. In *Computer Animation and Social Agents (CASA 2010)*, 2010.

[34] K. Pullen and C. Bregler. Motion capture assisted animation: Texturing and synthesis. *Proceedings of SIGGRAPH*, pages 501–508, 2002.

[35] A. Safonova. *Reducing the search space for physically realistic human motion synthesis*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, September 2006.