A Hybrid Approach for Simulating Human Motion in Constrained Environments

Jia Pan[♯], Liangjun Zhang[†], Ming C. Lin[♯] and Dinesh Manocha[♯]

panj@cs.unc.edu, zhanglj@cs.stanford.edu, lin@cs.unc.edu, dm@cs.unc.edu

[♯] Dept. of Computer Science, UNC Chapel Hill [†] Dept. of Computer Science, Stanford University

Abstract

We present a new algorithm to generate plausible motions for high-DOF human-like articulated figures in constrained environments with multiple obstacles. Our approach is general and makes no assumptions about the articulated model or the environment. The algorithm combines hierarchical model decomposition with sample-based planning to efficiently compute a collision-free path in tight spaces. Furthermore, we use path perturbation and replanning techniques to satisfy the kinematic and dynamic constraints on the motion. In order to generate realistic human-like motion, we present a new motion blending algorithm that refines the path computed by the planner with motion capture data to compute a smooth and plausible trajectory. We demonstrate the results of generating motion corresponding to placing or lifting object, walking and bending for a 38-DOF articulated model.

Keywords: Path planning, Motion synthesis, Motion capture and retargeting

Introduction

The problem of modeling and simulating human-like motion arises in different applications, including humanoid robotics, computer animation, virtual prototyping, and exploration of sensor-motor basis for neuroscience. This is a challenging problem due to both combinatorial and behavioral complexities. For example, the entire human body consists of over 600 muscles and over 200 bones and there are no known accurate and efficient algorithms to simulate their motion. Even the simplest human-like models that represent the skeleton as an articulated figure need at least $30-40$ joints to model different motions such as navigation, sitting, walking, running, object manipulation, etc. The high dimensionality of the configuration space of the articulated model makes it difficult to efficiently compute the motion. In addition to collision-free and kinematic constraints, we also need to ensure that the resulting trajectory satisfies the posture and dynamic constraints and looks realistic.

There exists extensive literature relevant to simulating human-like motion in robotics, biomechanics, animation and related areas. This includes motion planning algorithms that use randomized or sample-based methods to compute collision-free paths in the high-dimensional configuration space. However, the complexity of these algorithms increases significantly with the number of degrees-of-freedom (DOF) while prior methods are limited to low-DOF articulated figures e.g. less than 20 [1, 2, 3]. Most of the research in computer animation is based on motion capture, which tends to generate the most realistic human-like motion. Many techniques have been proposed to edit and modify or retarget the motion capture (mocap) data. However, it is difficult to capture the motion in constrained environments with multiple obstacles due to occlusion problems. Furthermore, it is hard to reuse or playback the motion in a virtual environment, which is different from the original environment in which the motion is captured [4, 5, 6, 7].

In this paper, we primarily focus on generating human-like motion in constrained environments with many obstacles, cluttered areas, or tight spaces. Aside from computer animation, such scenarios also arise in virtual prototyping, where modeling of digital humans or mannequins is used for design, or maintenance in CAD/CAM and ergonomic analysis [8, 9].

Main Results: We present an original hybrid approach that combines motion planning algorithms for high-DOF articulated figures with motion capture data to generate collision-free motion that satisfies both kinematic and dynamic constraints. Our approach performs whole-body planning by coordinating the motion of different parts of the body and later refines the trajectory with mocap data. The two novel components of our work include:

- **Decomposition planner:** In order to deal with high-DOF articulated figures, we use a hierarchical decomposition of the model and perform constrained coordination to generate a collision-free trajectory and maintain static/dynamic balancing constraints. The resulting planner computes the path for low-DOF components in an incremental manner and uses *path constraints* and *path perturbation* to generate a trajectory that satisfies kinematic and dynamic constraints.

- **Trajectory Refinement:** In order to overcome the random nature of sample-based planners and generate realistic motion, we refine the motion computed by our planner with mocap data to compute smooth paths. The resulting motion blending algorithm analyzes the motion and automatically builds a mapping between the path computed by the planner and the mocap data. We ensure that the resulting path still satisfies various constraints.

We demonstrate the results on generating human-like motion for a 38-DOF articulated model using the CMU mocap database. Our system can handle very cluttered environments to generate object grasping, bending, walking and lifting motions.

The rest of the paper is organized as follows. We first briefly survey prior work, followed by an overview of our approach and the planning algorithms. The motion generation and blending algorithms are described in the section titled ”Trajectory Refinement”. We highlight their performance and compare with prior approaches in the final section.

Related Work

The problem of simulating and generating human-like motion has been extensively studied in robotics, computer animation and biomechanics. In this section, we briefly survey prior methods on motion planning and animation.

Motion Planning

Sample-based approaches have been successfully applied to human-like robots to plan various tasks and motions. These include efficient planning algorithms for reaching and manipulation that combine motion planning and inverse kinematics [10, 11] or computing the whole body motion [1]. Moreover, motion strategies for human-like robots such as walking, sitting or jumping can also be computed by walking pattern generators [2, 3]. In order to plan collision-free and dynamically stable motions, many earlier approaches used a two-stage decoupled framework [12, 13, 14]. Task-based controllers have also been used to plan and control the whole-body motion [15, 16].

In order to generate natural-looking motion, many authors have proposed a two-stage framework. The planner first computes the motion taking into account a few or partial DOFs of the human model, e.g., cylindrical lower-body [17], manipulator [18], footsteps [19], etc. In the second stage, some motion data (e.g. mocap data) is retargetted by using the planned trajectory as constraints, e.g., a 2D trajectory [17, 19] or inverse-kinematics [18]. These methods typically work well in terms of generating regular motion (e.g. locomotion) in somewhat open environments without many obstacles.

Character Animation

There is extensive literature on motion generation in computer animation. At a broad level, prior methods can be classified into kinematic and dynamic methods. The basic kinematic methods use operators such as re-sequencing and interpolation to recombine the example motions into new motions, as is done in motion blending [20] and motion graphs [21]. Some recent variants [22] use optimization methods to satisfy the constraints. These methods can create natural long clips with a variety of behaviors, but their results are restricted within the linear space spanned by the example motions. Moreover, they need to be combined with global planning or collision avoidance schemes in order to handle constrained environments. Shapiro et al. [4] described an elegant approach to combine mocap data with motion planning.

Recently, dynamics-based methods have become popular in the animation community, e.g. [23, 24]. These approaches use a control strategy (e.g. PD control) to actuate a dynamics model (e.g. Newton-Euler equation). In practice, they are primarily used for interactive response and may not work well for collision-free motion computation in constrained environments.

Overview

In this section, we present an overview of our approach on generating natural human motion in constrained environments. The overall pipeline of our algorithm is given in Fig 1(a). We do not make any assumptions about the environment or the obstacles in the scene. We assume that the human-like model is represented by an articulated model with serial and parallel joints and there is no limit on the number of DOFs.

Our approach first uses a sample-based high-DOF planner to compute a collision-free path, and it also takes into account the foot placement constraint and static/dynamic balancing constraints. In order to deal with a large DOFs, we use a hierarchical decomposition scheme and present an efficient decomposition planner.

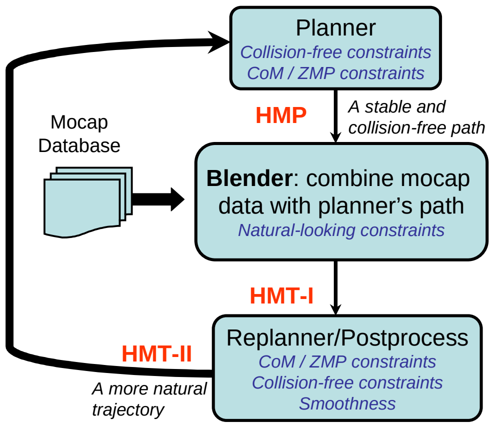
In practice, generating natural-looking motion using planning algorithms is considered non-trivial, due to the following reasons: firstly, the randomness of the motion planner can cause jerky and unnatural motion, especially when parts of the robot are in open space; secondly, computing a collision-free trajectory corresponds to searching in a very high-dimensional space, which can be quite challenging in computation; finally, the constrained environments may have tight spaces or narrow passages and this makes it hard for even sample-based planners to search for a valid trajectory. In order to address these issues, we present a novel motion blending algorithm, which refines the motion computed by the planner with motion capture data. We also ensure that the resulting motion is collision-free and satisfies all the other constraints.

Decomposition Motion Planner

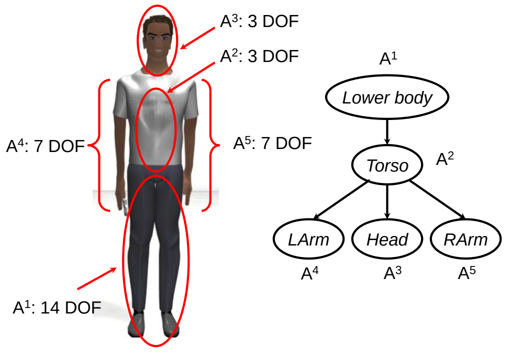
A human-like model is a tightly coupled high-DOF system and the inter-connections between the body parts must be maintained during motion planning. We use a constrained coordination scheme to find a collision-free path, which is based on prior work on *incremental coordination* [25, 26]. Instead of performing a whole-body planning in a high-dimensional configuration space, our planner computes the collision-free path using incremental steps. First, we decompose the model into n components $\{A^1, ..., A^n\}$ and each component has $|A^i|$ joints. Usually such a decomposition scheme corresponds to different parts of the human body that have relatively low-correlation with each other. For example, in Fig 1(b) the model has 5 components. Next the planner computes the collision-free path incrementally in n steps: in the k-th step, it computes a collision-free path M_k for sub-model $\breve{A}^k \equiv \bigcup_{i=1}^{k} A^i$. In the first step, collision-free path M_1 for A^1 is computed by standard RRT algorithm, a sample-based approach. For other steps, we apply the *path constraint* bias on the sub-model $\breve{A}^k = \breve{A}^{k-1} \bigcup A^k$: only A^k can sample its joint values randomly, while the configuration of $\breve{A}^{k-1}$ must be constrained on the one-dimensional space M_{k-1}, which is the collision-free path obtained by the last step. Such sampling is called *constrained sampling* [27]. As a result, the planning dimension is reduced from $\sum_{i=1}^{k} |A^i|$ to $1 + |A^k|$ and we use the RRT planner to find a collision-free path for $\breve{A}^k$ in this lower-dimensional space. We further use retraction-based technique to improve planner's behavior for narrow passages in constrained environments [28].

The final step returns a collision-free path for the whole body.

Our decomposition strategy has several advantages. First, we can solve a high-dimensional planning problem by solving a series of lower-dimensional sub-problems. Therefore it can provide significant acceleration in many scenes. Secondly, the *path constraint* can reduce the jerkiness caused by random sampling in the high-dimensional space. Finally, it fits well with human's behavior in constrained spaces: though different components of the human body can be highly coupled for motions in open environments (e.g. free walking), their correlation with one another is lower when part or all of the human body is in constrained scenes.


Planner
Collision-free constraints
CoM / ZMP constraints
HMP
A stable and collision-free path
Mocap Database
Blender: combine mocap data with planner's path
Natural-looking constraints
HMT-I
HMT-II
Replanner/Postprocess
CoM / ZMP constraints
Collision-free constraints
Smoothness
A more natural trajectory

(a)



(b)

Figure 1: (a) An overview of our hybrid approach, which can combine the motion computed by planner and the motion from mocap databases to generate a collision-free, dynamic and natural human motion. (b) Our 5-component decomposition scheme for a 38-DOF human-like model. We compute a trajectory for each component in an incremental manner.

The decomposition strategy works well in most cases. However, the *path constraint* bias is based on a greedy formulation, and the overall algorithm may fail to compute a collision-free path in some cases. For example, when $\breve{A}^{k-1}$ is constrained to moving along path M_{k-1}, it can be viewed as a moving obstacle for A^k and may create a narrow passage or even block A^k. In order to handle such problems, we use *path perturbation* [27] to search for a collision-free path in the local neighborhood or use a different decomposition scheme.

Motion Constraints

The underlying coordination algorithm can be extended to generate a statically stable motion. At each incremental step of the high-DOF motion planner, we check whether a configuration $\mathbf{q}$ generated from the constrained sampling is statically stable, i.e., the projection of the center of mass (CoM) point of the model lies inside the support polygon defined by the support feet of the human. If $\mathbf{q}$ is not statically stable, we perturb it locally to generate a statically stable configuration $\mathbf{q}'$ and also ensure the foot placement is not changed. As the CoM can be locally represented as a linear function of $\mathbf{q}$, we perform this perturbation by reducing to an inverse kinematic (IK) algorithm which tries to move the CoM towards the center of the support polygon. To maintain the foot contacting constraint, contact feet are added as additional end-effectors, and in IK formulation their positions and orientations must remain unchanged.

The modified constrained sampling allows us to generate statically stable samples that are used to construct a connectivity roadmap in the free space of the articulated model. We also need to check whether the interpolating motion between two nearby samples (i.e. computed by the local planning step) is also a statically stable motion. One simple way of doing this is to generate many discrete samples along the interpolating motion and check each of these samples individually. If any of the samples is not statically stable, we perturb it using our IK-based CoM perturbation algorithm. Ultimately, we ensure that the path computed by the planner is collision-free and satisfies the stability constraints.

Once we obtain a statically stable path, we can modify it to ensure that it is also dynamically stable. First, for each configuration $\mathbf{q}$, we estimate its velocity $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ from motion capture data with the approach that will be introduced in the next section. Then, we calculate the *zero moment point* (ZMP) and check whether it lies within the support polygon of the human feet. If not, we perturb $\mathbf{q}$ to generate a dynamically stable configuration. Similar to CoM, ZMP can also be represented as a linear function of $\mathbf{q}$, if $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are fixed. Therefore, such perturbation can also be performed based on the IK method. This method works well for our kinematics-based planner when the original ZMP does not deviate too far away from the support polygon. For cases when ZMP deviates a great deal from the support polygon, global path optimization such as [29] can be used to guarantee dynamic balancing.

Motion Blender

The output of the high-DOF planner is a collision-free path M, we call it the *Human Motion Path* (HMP). The decomposition planner only considers collision-avoidance constraints and uses random sampling. This approach can lead to a jerky motion along the trajectory. As a result, we augment or modify HMP by using mocap data, if available.

We process the postures in HMP in a per-component manner. First we analyze the postures in HMP based on criteria which takes into account the local environment (i.e. nearby obstacles) around the posture and the quality of the computed path.

Based on these criteria, we process different components of each posture with different strategies: 1) For components that lie in a constrained space, we primarily rely on the samples in HMP, even though the computed path may not be natural-looking. In these cases, the planner computes a collision-free and statically stable path, and any large changes to that path may result in collisions. As a result those postures of HMP are used in the final path, we only allow small perturbations during the refinement. 2) For components of the human-model that lie in open space and appear to be natural, we tend to retain those postures. Otherwise, we compute a configuration based on the mocap database. The output of this phase is a trajectory that combines HMP with the mocap data and we refer to it as *Human Motion Trajectory I* (HMT-I). HMT-I may not be collision-free or even smooth, but it contains some important information from the mocap data that can bias the result of the motion planner towards a natural-looking trajectory. We finally perform a decomposition-based replanning with HMT-I as the guidance path, and compute a collision-free trajectory called HMT-II.

Trajectory Refinement

In this section, we present our *motion blending* algorithm. It starts with the collision-free path HMP and refines the path based on the mocap data to improve its smoothness and to generate more natural-looking motion. Fig 2 shows an overview of different modules used within the motion blender: the *search and mapping* module searches the mocap library to find motion clips that match best with HMP and builds a mapping between them; the *criteria* module decides when to use the mocap data for refinement; *mocap inference* and *perturbation* modules deal with issues related to blending the mocap data with the postures in HMP; and the *replanning* module uses all this information to generate a smooth and collision-free trajectory, HMT-II.

We first introduce notations used in the remaining part of this section. Symbol M

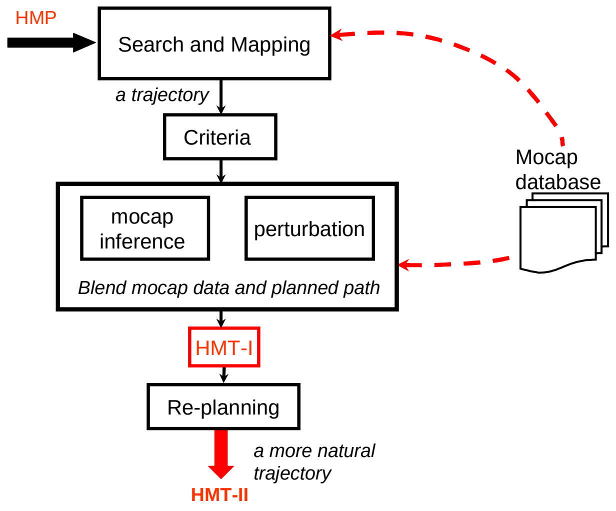


Figure 2: An overview of the *motion blending* algorithm.

represents the input human motion path (HMP) computed by the decomposition planner. $\mathbf{q}(t) = M(t)$ is one node on the path and t is the parameter along the path. We also use $\mathbf{q}(t)$ as the configuration of that node. Due to model decomposition, $\mathbf{q}(t) = \{\mathbf{q}^1(t), ...\mathbf{q}^n(t)\}$, where $\mathbf{q}^i$ is the configuration for A^i. For convenience, we usually use $\mathbf{q}$ to represent the *current* path node.

Search and Mapping

In this step, we first search the mocap database for a motion clip that matches best with the path M. Next, we construct a one-to-one mapping between M and one segment C of that clip. In general, M is a continuous path, but has no time parametrization associated with it. This mapping will associate the time information from the mocap data with the planner data. In the *criteria* module, the timing information encoded in the mapping is used to compute torque variation criteria. And in the *motion inference* step, mapping can provide natural candidate postures for low-quality planning results.

We use a simple scheme to perform the search step. For each posture of M and a posture of a certain motion clip in the database, we compute the distance between them, using the horizontal translation- and vertical rotation-invariant metrics [21]. Mocap clips with sufficient numbers of matching frames, i.e. with a distance smaller than certain threshold, are selected as the potential matching clips. In these clips, we find segments whose two ends match at the two ends of M. For all remaining segments, we compute a time-wrapping function f that minimizes the following energy function: $E = \sum_{t=0}^{|M|-1} \|M(t) - C(f(t))\|_{MG}$, here $|\cdot|$ is the number of nodes on the path and $\|\cdot\|_{MG}$ is horizontal translation- and vertical rotation-invariant metric.

We require a valid time-wrapping function f to satisfy 3 constraints. 1) Boundary constraint: endpoints of $M(t)$ and $C(t)$ must match, i.e. $f(0) = 0$ and $f(|M|) = |C|$; 2) Monotonic constraint: $f(t) \leq f(t+1)$, which guarantees a time-wrapping function is invertible. 3) Slope constraint: a time wrapping function should not be too steep or too shallow: $L_l \leq f(t+1) - f(t) \leq L_u$, where $L_l = 0.8\frac{|C|-1}{|M|-1}$ and $L_u = 1.2\frac{|C|-1}{|M|-1}$. Our mapping method is similar to the registration algorithm in [30] except that we require the additional boundary constraint. We also reduce the optimization into a non-linear fitting problem and use the Levenberg-Marquardt algorithm to solve it.

Given all the potential segments, we select the best-matching segment C as the one whose distance from M, $\sum_{t=0}^{|M|-1} \|M(t) - C(f(t))\|_{MG}$, is the smallest. Based on the matching, we associate the time parametrization of C with M and represent it as a time-varying trajectory.

Blend Criteria

A key component of our approach is to decide which postures of HMP need to be replaced or augmented with the mocap data. We use three heuristics: space clearance (CLR), posture similarity (PS), and torque variation (TV). All three are per-component criteria, i.e., are computed for each component A^i of a human-like model, reason being that in a constrained environment the postures computed by the motion planner usually appear unnatural in some parts of the model. We hope to use the configuration of natural parts and only adjust those unnatural components. We assume that the model's configuration is $\mathbf{q} = \{\mathbf{q}^1, ..., \mathbf{q}^n\}$, where n is the number of components.

Some of the prior methods use data-driven based quantified measurements [31] to formulate appropriate criteria. However, such an approach may not be applicable for constrained environments. First, the models in [31] are trained on some regular motions (e.g. walking) or high-dynamic motions (e.g. dancing). Those models may not be able to classify natural and unnatural motions in constrained environments. Secondly, their basic assumption is 'motions that we have seen repeatedly are judged natural,' which is correct for motions in open spaces. However, in a constrained space, humans may behave differently to avoid obstacles, which may produce some variation not included in the mocap database. Finally, the prior models measure the naturalness for the entire body rather than each component. Instead, we propose some different criteria to evaluate the motion of different components.

Space Clearance

Space clearance evaluates whether a component A^i is in the constrained environment or not. To compute this, we first fix $\mathbf{q}^j$ for all $j \neq i$. Then we generate samples in the neighborhood of $\mathbf{q}^i$ uniformly. We check whether each sample collides with any obstacles in environments and use $\text{CLR}(\mathbf{q}^i) = \frac{\#\text{non-collision samples}}{\#\text{all samples}}$ as a metric to estimate the space clearance, while $\#(\cdot)$ is the counting function. In other words, we compute the possibility that a small variation of $\mathbf{q}^i$ will produce an in-collision configuration, which implicity describes the local distribution of obstacles near A^i's current position. If $\text{CLR}(\mathbf{q}^i)$ is larger than a given threshold (e.g. 50%), we estimate that the component A^i is in the open free space. Otherwise, we estimate that this posture is in the constrained space, i.e., close to the boundary of the free space.

Posture Similarity

Another important criterion is to evaluate whether the current configuration $\mathbf{q}^i$ of A^i is similar enough to ascertain $\mathbf{q}_d^i$ in the posture database that is generated as part of a preprocess (see next section). The posture distance for a component is simply defined as the squared distance of joint angles: $\|\mathbf{q}^i - \mathbf{q}_d^i\|_2$. However, such simple distance can not evaluate the natural aspect of motion well. The reason is that natural motion interpretation is relative with respect to other components: even if the configuration of a component is fixed, its visual impact also changes when the configurations of other body components change. Our solution is based upon the hierarchical representation of the human-like models in Fig 1(b). We notice that such dependency between the components is strong only for adjacent components in the hierarchical representation. We use a conditional distance metric $\|\mathbf{q}^i - \mathbf{q}_d^i\|_C = \begin{cases} \|\mathbf{q}^i - \mathbf{q}_d^i\|, & \|\mathbf{q}^j - \mathbf{q}_d^j\| \leq \epsilon \\ \infty, & otherwise \end{cases}$ to measure how natural-looking A^i motion is, where A^j is the parent component of A^i in the component hierarchy. With the $\|\cdot\|_C$ metric, two postures are similar in one body component A^i only if they are similar in the parent component A^j as well. We can then define the criteria for natural-looking motions as: $\text{PS}(\mathbf{q}^i) = \max_{\mathbf{q}_d \in database} \exp^{-\frac{\|\mathbf{q}^i - \mathbf{q}_d^i\|_C}{\sigma^2}} \omega(\mathbf{q}^i)$, where $\omega(\mathbf{q}^i) = \frac{\sum_{\|\mathbf{q}_d^i - \mathbf{q}^i\| \leq \epsilon_1, \|\mathbf{q}_d^j - \mathbf{q}^j\| \leq \epsilon_2} \#\mathbf{q}_d}{\sum_{\|\mathbf{q}_d^j - \mathbf{q}^j\| \leq \epsilon_2} \#\mathbf{q}_d}$ is the weighting function, which measures the ratio of the frequency of database postures similar to $\mathbf{q}$ in both A^i and its parent A^j to the frequency of those postures similar to $\mathbf{q}$ only in parent component A^j. If $\omega(\mathbf{q}^i)$ is small, it implies that the current configuration of A^i is not so common or frequent in the database, so we assign it a smaller weight. If $\text{PS}(\mathbf{q}^i)$ is larger than the threshold, then A^i is regarded to be in its natural configuration.

Torque Variation

We have shown how M can be associated with timing information (i.e. a parameterization). Therefore, we can estimate $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ for each point on M. Then we can compute the torque τ for human body by inverse dynamics.

In a natural-looking motion, the torque of each joint tends to change gradually and this boils down to minimizing the integral $\int_0^T |\dot{\tau}|^2 dt$. This formulation has also been used as a motion constraint in [24]. Using calculus of variation, this means $\ddot{\tau} = 0$, i.e., τ should changes linearly between $\tau(\mathbf{q}_s)$ and $\tau(\mathbf{q}_g)$, where $\mathbf{q}_s$ and $\mathbf{q}_g$ are the first and last configuration of HMP. When $\tau(t)$ deviates from the linear formulation, it may result in a more unnatural motion. Thus we define the torque criteria as $\text{TV}(\mathbf{q}^i) = \max_{j \in A^i} \frac{|\tau_j(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) - \text{Lerp}(\tau_j(\mathbf{q}_s), \tau_j(\mathbf{q}_g), t)|}{|\text{Lerp}(\tau_j(\mathbf{q}_s), \tau_j(\mathbf{q}_g), t)|}$, where τ_j is the torque for joint j and $\text{Lerp}(\cdot)$ is the linear interpolation function. If $\text{TV}(\mathbf{q}^i)$ is larger than a given threshold, we disregard those motions.

Blend Strategy

After all three criteria highlighted in the previous section are computed, we use the results to perform the *motion blending* algorithm. Our approach works as follows. We use motion inference only when the segment of path is in open environment and does not appear to be natural. This happens when CLR($\mathbf{q}^i$) $> T_1$ AND (PS($\mathbf{q}^i$) $< T_2$ OR TV($\mathbf{q}^i$) $> T_3$). Then we conclude that $\mathbf{q}^i$ should not be used and we rather use a better configuration from the mocap database. Otherwise, we perform some perturbation around $\mathbf{q}^i$. Here T_1, T_2, T_3 are three thresholds.

The perturbation operation is simple: for $\mathbf{q}^i$, we first find a $\mathbf{q}^i_*$ in the motion database that minimizes the distance to it: $\mathbf{q}^i_* = \mathrm{argmin}_{\mathbf{q}_d \in DB} \|\mathbf{q}^i_d - \mathbf{q}^i\|_C$. This nearest mocap data is interpolated with $\mathbf{q}^i$ to get the following perturbation result: $\tilde{\mathbf{q}}^i = \mu \mathbf{q}^i + (1-\mu)\mathbf{q}^i_*$, where $\mu = \exp^{-\|\mathbf{q}^i_* - \mathbf{q}^i\|^2_C/\sigma^2}$ for $\mathbf{q}^i$ in open environment and $\mu = 1 - \exp^{-\|\mathbf{q}^i_* - \mathbf{q}^i\|^2_C/\sigma^2}$ for $\mathbf{q}^i$ in constrained environment. Such difference about interpolation weight reflects the fact that in constrained space, the collision-free aspect of motion is more important, while in open space natural-looking motion generation is more important.

The inference step tends to be more complicated. As $\mathbf{q}(t) = M(t)$ is no longer a good configuration for the final path, we compute a better configuration based on the mapped mocap data $\mathbf{q}_m(t) = C(f(t))$ and path smoothness. The new configuration $\tilde{\mathbf{q}}^i(t)$ for component A^i is given as $\tilde{\mathbf{q}}^i(t) = \underset{\mathbf{q}_d \in DB}{\mathrm{argmin}} \|\mathbf{q}^i_d - \mathbf{q}^i_m(t)\|_C + \lambda \|\mathbf{q}^i_d - \tilde{\mathbf{q}}^i(t-1)\|_C$. In this expression, the first term minimizes the conditional distance to the mapped mocap data. The second term minimizes the conditional distance to the perturbed result of the last time step. As our algorithm always tries to find a natural-looking posture for every node on HMP, the perturbed or inferred posture of the last time step has a high probability of generating natural looking motion, which can provide additional information for the current posture and can improve the smoothness of inferred motion $\tilde{\mathbf{q}}^i(t)$.

The output of *motion blending* algorithm is represented as HMT-I: some postures of it comes from HMP while other portions of that trajectory are generated from the mocap inference. However, it is not guaranteed to be collision-free or even smooth, so we use replanning to satisfy the collision-free hard-constraint.

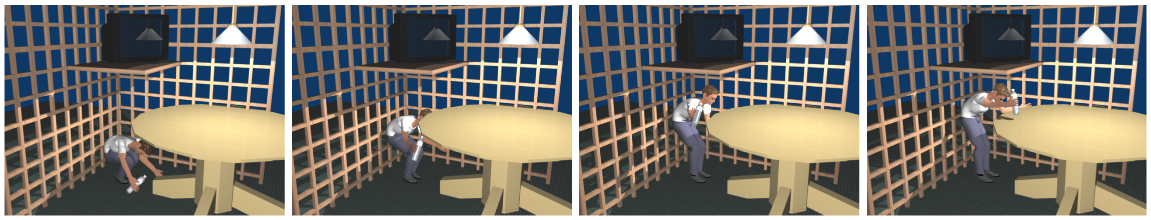


Figure 3: Object Retrieval: In this benchmark, the human model stands up and places the object on the table. The overall motion involves use of many DOFs and the resulting path is collision-free.

Replanning

In this step, the motion planner performs replanning by using HMT-I as the starting trajectory and refining it. The final trajectory, HMT-II, is guaranteed to be collision-free and continuous. In robotics, replanning [32] is generally used for scenarios when the information available about the environment is incomplete or the environment is dynamic with moving obstacles. The replanning algorithm can update a solution using incremental methods, as opposed to computing a new collision-free path from scratch. Our current system is mainly limited to static scenes, and so we use replanning to include naturalness bias into the planner and fix some of the issues in the trajectory HMT-I. The difference between our replanner and that in [32] is that we bias the path for natural-looking and smooth motions.

Our replanner also uses the framework of a decomposition-based planner [27]. The only difference is that we change the sampling function: with a probability of p_{follow}, the planner samples within the 1-dimension space of HMT-I and with a probability of $1-p_{follow}$, the planner generates samples in a higher-DOF space. The sampling bias for nodes on or near HMT-I can encode natural-looking constraints implicitly in the replanning process. The output of the replanner is a collision-free trajectory HMT-II with high quality. It may still have small visual artifacts, but these can be fixed by using local smoothing [18] and perturbation in postprocessing.

Implementation and Performance

In this section, we describe some implementation details and highlight the performance of our algorithm on many challenging benchmarks.

Benchmarks

We designed three challenging environments with many obstacles and tested our approach to generate collision-free motion by specifying the initial and final configurations of the human model. All the benchmarks consist of multiple obstacles and it would be difficult to edit mocap data directly for such settings. Rather we generated the initial path (HMP) using our planner and used some postures from the CMU mocap database to make the motion appear to be more natural (as shown in the video). In order to generate the appropriate motion, we needed to utilize the multiple DOFs of the human-model. It is rather difficult to generate or simulate such motions using a low-DOF planner.

In the first benchmark (Object Retrieval, Fig 3), human begins from a knee-bending posture, tries to pick up an object and then puts it on the table. There are three main obstacles: grate behind, a ceiling above and the table. The ceiling and table pose as the obstacle constraints. When the human is bending down, the motions of its hands and head are constrained by the table and when the human is standing up, its hands are in the open environment, but the head is restricted by the ceiling object. The round edge of table also leads to a difficult narrow passage in the configuration space. Another aspect makes the benchmark even more challenging: when under the table, the right limb encounters obstacles in all directions except to the right. The sample-based planner tends to avoid obstacles and so the right limb will go right first instead of going up directly, which would cause an unnatural, big-rotation motion.

The second benchmark is the manipulation task (Object Placement Fig 4). In this case, the human holds an object (e.g. a book) and rotates backward to put it on a shelf. The grate, lamp, and bookshelf result in a tight and constrained environment.

The third benchmark has two main motion components: walking and bending (Fig 5).

	Object Retrieval (Fig. 3)	Object Placement (Fig. 4)	Walking & Bending (Fig. 5)
#obstacles	4	5	7
#polygons	7967	52810	372609

Table 1: Geometric complexity of our benchmarks.

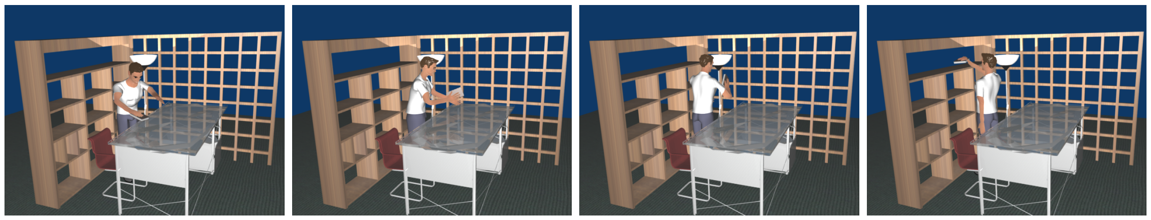


Figure 4: Object Placement: The human model picks the book from the table and puts it on the bookshelf.

The human first walks along a passage with an obstacle around its head and then puts a tool into the car. The task of putting the object inside the car is challenging due to limited space. Moreover, the planner again has several potential paths to put hands into the car and the shortest path is not natural. Fig 6 show a more challenging variation of this benchmark with more human bending.

Implementation

A key aspect of our hybrid algorithm is to ensure that the underlying mocap data used to augment the motion has a similar kinematic representation as the underlying human-model used in our system. In our approach, we use CMU's motion capture library and ensure that the joint structure matches with our human-model. If the underlying motion data in the library is different, we transform that into our joint structure. Usually the capture frequency of mocap data is rather high, therefore we compress the data for a more compact posture database. For efficiency, we use the greedy clustering algorithm similar to [18], but the clustering centers are stored in terms of some unique joint angles rather than 3D positions of markers in [18]. We select joint angles instead of marker positions because we always need to compute the similarity between components from mocap database and our human model.

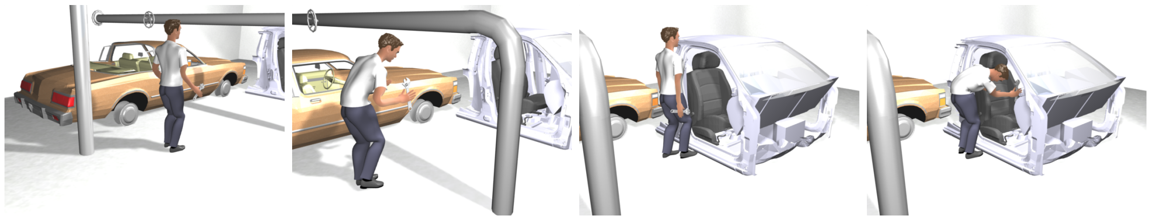


Figure 5: Walking & Bending: The human model walks towards the car, avoiding some obstacles. It bends and stretches to put the tool inside the car. The latter is a highly constrained environment and we need to perform whole-body planning to generate such motions.

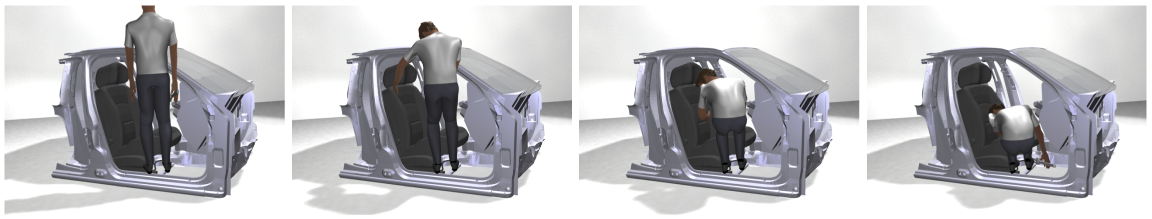


Figure 6: More bending: The human bends and stretches to put the tools under the chair. This environment is far more constrained than the one of Walking & Bending (Fig. 6). The whole-body planning becomes even more challenging in such constrained environments.

Usually, the main body component in the mocap data does not have many markers associated with it and it can be hard to estimate the alignment parameter between postures using the marker position-based method [21]. Moreover, we also need to consider the configurations of its parent component. For each posture $\mathbf{q}_d$ in the database, we also record the number of postures it represents during compression, as $\#\mathbf{q}_d$. This was used to compute the posture similarity (PS) criterion.

In terms of the blend criteria, torque variation (TV) and posture similarity (PS) can be computed very fast. The computation of clearance (CLR) is slower, because it needs to generate about 1000 samples and performs collision checking with the environment.

Results and Analysis

Table 2 shows the runtime performance of our algorithm. The clearance estimation module takes most of the time due to a high number of samples and collision checking. Decreasing

	planning	search&mapping	PS	TV	CLR	blend	replanning	sum
Object Retrieval	0.339	0.188	0.516	4.235	26.79	2.844	0.412	35.32
Object Placement	2.402	0.188	0.625	3.157	28.95	3.578	4.930	43.83
Walking & Bending	9.765	6.419	0.779	6.784	30.67	10.52	14.53	79.46

Table 2: Time for each module of our algorithm (in sec).

the number of samples can improve its efficiency.

We have shown the trajectories computed by HMP and HMT-II for some of the benchmarks in the video. In order to highlight the improvement of HMT-II over HMP quantitatively (see Fig 7) we show how the torque variation and posture similarity change over time on the trajectory for each of them. In this scenario, the right limb is in the open environment and HMP has quite a low PS value. The left limb is in the constrained environment and HMP can produce acceptable motion with a larger PS value. HMT-II shows large PS values in both cases. Similar results occur for torque variation: HMP's torque variation curve deviates considerably from linearity (high TV value) for the right elbow but fits a straight line quite well (low TV value) for the left elbow. HMT-II's curves have a low TV value in both cases. This shows that HMT-II indeed provides higher quality compared with the motion planner result HMP.

Comparison

The notion of combining motion planning with mocap data for generating human-like motion is not new. Some prior methods, including the work of Yamane et al. [18] and Shapiro et al. [4], use such combinations. Our framework is similar to that of [18]: use first motion planning and then mocap refinement. Similar to their method, we consider obstacle avoidance to be our primary constraint and generating natural-looking motion as secondary. Yamane et al.'s method may not work well in highly constrained environments. After computing a collision-free path for manipulated objects, it directly uses IK biased by mocap data to reconstruct the whole-body motion. This approach implicitly makes three assumptions: 1) the IK algorithm can compute a collision-free whole-body configuration; 2) There exists a

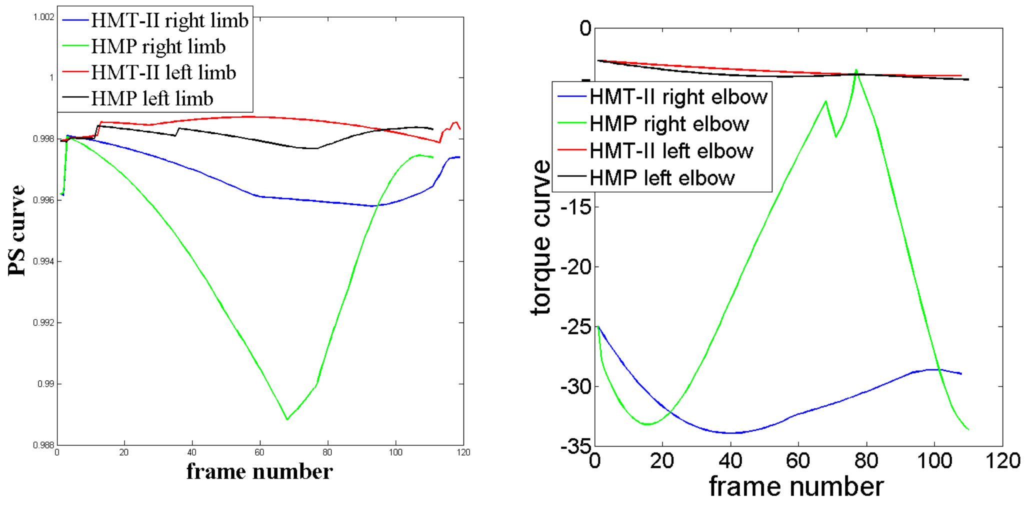


Figure 7: Comparison of posture similarity (PS, left) and torque variation (TV, right) between HMP and HMT-II, in the Object Retrieval benchmark.

collision-free path to locally connect every two neighboring configurations reconstructed by IK. 3) The human motion required for any given scene are included in the mocap database and the resulting motion is more natural looking. All of these requirements are often difficult to satisfy in constrained environments (e.g. our benchmarks). In contrast, our approach can easily handle very high-DOF articulated models, can search for valid motions in narrow passages, and can be integrated with somewhat sparse mocap databases.

Shapiro et al. [4] uses a dual framework: they first use mocap data and when necessary apply motion planning algorithm to avoid obstacles. This method places a higher priority on natural-looking motion as opposed to collision-free motion and other constraints. In practice, this approach is better suited for open scenarios with a few obstacles and not the constrained environments with tight spaces. Furthermore, [4] uses a RRT planner for whole-body planning and this approach can become expensive for high-DOF articulated models. In many ways, our approach can be regarded as a dual of this method and is targeted for different scenarios.

Conclusion

We have presented an algorithm that combines a high-DOF motion planning algorithm with mocap data to generate plausible human motion and satisfy geometric, kinematic and dynamic constraints. We use a hierarchical decomposition of a high-DOF articulated model and use that decomposition for constrained coordination and to satisfy different constraints. We also present automated techniques to search a mocap database for plausible motion clips and generate a smooth, blended motion. We highlight the performance on generating motion for different tasks including object placement, object retrieval and walking&bending.

Our approach has some limitations. The underlying planner uses a randomized or sample-based scheme to compute configurations in the free space and join them using local planning. If there are narrow passages in the free space, the underlying approach may not be able to compute a path that satisfies all the constraints or may take a very long time. Our approach assumes that the underlying mocap database has sufficient samples to generate realistic motions. The searching strategy used to find the best-match mocap clip is based on minimizing an energy function and its performance can vary based on the path (HMP) computed by the planner and the mocap data. The underlying criteria for motion refinement are based on three heuristics and can also fail in some cases.

There are many avenues for future work. We can further improve the search and mapping algorithms and the various criteria used by our *motion blender*. Moreover, it may be useful to also consider non-holonomic and other constraints [5] in our planner to improve the accuracy and realism of the HMP.

Acknowledgements. We would like to thank Jean Paul Laumound and Kineo for providing car models. We would like to thank Sean Curtis and Will Moss for their help in video and rendering. This research was supported in part by ARO Contract W911NF-04-1-0088, NSF awards 0636208, 0917040 and 0904990, DARPA/RDECOM Contract WR91CRB-08-C-0137, and Intel. Liangjun Zhang was supported in part by CRA/NSF/CCC Computing Innovation Fellows Project.

References

[1] Kris Hauser, Timothy Bretl, Kensuke Harada Jean-Claude Latombe, and Brian Wilcox. Motion planning for legged robots on varied terrain. *International Journal of Robotics Research*, 27(11-12):1325–1349, Nov-Dec 2008.

[2] Qiang Huang, Kazuhito Yokoi, Shuuji Kajita, Kenji Kaneko, Hirohiko Arai, Noriho Koyachi, and Kazuo Tanie. Planning walking patterns for a biped robot. *IEEE Transactions on Robotics and Automation*, 17:280–289, 2001.

[3] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. *International Conference on Robotics and Automation*, pages 1620–1626, 2003.

[4] Ari Shapiro, Marcelo Kallmann, and Petros Faloutsos. Interactive motion correction and object manipulation. In *Symposium on Interactive 3D Graphics*, pages 137–144, 2007.

[5] Julien Pettre, Marcelo Kallmann, and Ming C. Lin. *Motion Planning and Autonomy for Virtual Humans*. ACM SIGGRAPH Course Notes, 2008.

[6] Marcelo Kallmann, Amaury Aubel, Tolga Abaci, and Daniel Thalmann. Planning collision-free reaching motions for interactive object manipulation and grasping. *Computer Graphics Forum*, 22(3):313–322, September 2003.

[7] Maciej Kalasiak and Michiel van de Panne. A grasp-based motion planning algorithm for character animation. *The Journal of Visualization and Computer Animation*, 12(3):117–129, 2001.

[8] Onan H. Demirel and Vincent G. Duffy. Applications of digital human modeling in industry. In *Digital Human Modeling, Lecture Notes in Computer Science*, volume 4561, pages 824–832. Springer Berlin / Heidelberg, 2007.

[9] Jean-Paul Laumond, Etienne Ferre, Gustavo Arechavaleta, and Claudia Esteves. Mechanical part assembly planning with virtual mannequins. In *International Symposium on Assembly and Task Planning*, pages 132–137, 2005.

[10] Rosen Diankov, Nathan Ratliff, Dave Ferguson, Siddhartha Srinivasa, and James Kuffner. Bispace planning: Concurrent multi-space exploration. In *Robotics: Science and Systems*, pages 159–166, 2008.

[11] Evan Drumwright and Victor Ng-Thow-Hing. Toward interactive reaching in static environments for humanoid robots. In *International Conference On Intelligent Robots and Systems*, pages 846–851, 2006.

[12] Kensuke Harada, Shizuko Hattori, Hirohisa Hirukawa, Mitsuharu Morisawa, Shuuji Kajita, and Eiichi Yoshida. Motion planning for walking pattern generation of humanoid. In *International Conference on Robotics and Automation*, pages 4227–4233, 2007.

[13] James Kuffner, Satoshi Kagami, Koichi Nishiwaki, Masayuki Inaba, and Hirochika Inoue. Dynamically-stable motion planning for humanoid robots. *Autonomous Robots*, 12(1):105–118, January 2002.

[14] Eiichi Yoshida, Igor Belousov, Claudia Esteves, and Jean-Paul Laumond. Humanoid motion planning for dynamic tasks. *International Conference on Humanoid Robots*, pages 1–6, 2005.

[15] Michael Gienger, Christian Goerick, and Edgar Körner. Whole body motion planning - elements for intelligent systems design. *Zeitschrift Künstliche Intelligenz*, 4:10–15, 2008.

[16] Luis Sentis and Oussama Khatib. Whole-body control framework for humanoids operating in human environments. *International Conference on Robotics and Automation*, pages 2641–2648, 2006.

[17] Julien Pettré, Jean-Paul Laumond, and Thierry Siméon. A 2-stages locomotion planner for digital actors. In *Symposium on Computer Animation*, pages 258–264, 2003.

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

[19] Min Gyu Choi, Jehee Lee, and Sung Yong Shin. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Trans. Graph.*, 22(2):182–203, 2003.

[20] Lucas Kovar and Michael Gleicher. Flexible automatic motion blending with registration curves. In *Symposium on Computer Animation*, pages 214–224, 2003.

[21] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Trans. Graph.*, 21(3):473–482, 2002.

[22] Alla Safonova and Jessica K. Hodgins. Construction and optimal search of interpolated motion graphs. *ACM Trans. Graph.*, 26(3):106, 2007.

[23] Yeuhi Abe and Jovan Popović. Interactive animation of dynamic manipulation. In *Symposium on Computer Animation*, pages 195–204, 2006.

[24] Sumit Jain, Yuting Ye, and C. Karen Liu. Optimization-based interactive motion synthesis. *ACM Trans. Graph.*, 28(1):1–12, 2009.

[25] Pekka Isto and Mitul Saha. A slicing connection strategy for constructing prms in high-dimensional cspaces. *International Conference on Robotics and Automation*, pages 1249–1254, 2006.

[26] Mitul Saha and Pekka Isto. Multi-robot motion planning by incremental coordination. In *International Conference On Intelligent Robots and Systems*, pages 5960–5963, 2006.

[27] Liangjun Zhang, Jia Pan, and Dinesh Manocha. Motion planning of human-like robots using constrained coordination. In *International Conference on Humanoid Robots*, 2009.

[28] Jia Pan, Liangjun Zhang, and Dinesh Manocha. Retraction-based rrt planner for articulated models. In *International Conference on Robotics and Automation*, 2010, to appear.

[29] Satoshi Kagami, Fumio Kanehiro, Yukiharu Tamiya, Masayuki Inaba, and Hirochika Inoue. Autobalancer: An online dynamic balance compensation scheme for humanoid robots. In *Workshop on Algorithmic Foundations of Robotics*, pages 692–698, 2000.

[30] Yen-Lin Chen, Jianyuan Min, and Jinxiang Chai. Flexible registration of human motion data with parameterized motion models. In *Symposium on Interactive 3D Graphics*, pages 183–190, 2009.

[31] Liu Ren, Alton Patrick, Alexei A. Efros, Jessica K. Hodgins, and James M. Rehg. A data-driven approach to quantifying natural human motion. *ACM Trans. Graph.*, 24(3):1090–1097, 2005.

[32] David Ferguson, Nidhi Kalra, and Anthony Stentz. Replanning with rrts. In *International Conference on Robotics and Automation*, pages 1243–1248, 2006.