2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Efficient Exact Collision-Checking of 3-D Rigid Body Motions using Linear Transformations and Distance Computations in Workspace

Liang He Jur van den Berg

***Abstract*— This paper presents a new method for efficient and exact collision-checking of linear motions of 3-D rigid bodies. 3-D rigid bodies have 6-D configuration spaces (three degrees of freedom for position and three for orientation), and constitute an important subclass of motion planning problems. Our method can be used with any collision-checker that is capable of performing linear transformations and distance computations on 3-D geometry. As previous work has shown, computing the distance between the rigid body in some configuration and the workspace obstacles immediately determines the collision-status of surrounding configurations. Using a recursive procedure one can then determine exactly whether an entire motion of the rigid body is collision-free. In this paper, we will show that by performing an optimally selected *linear transformation* on the workspace, the collision-status of rigid body motions can be determined using significantly fewer (costly) distance computations. Since collision-checking is often the computational bottleneck in sampling-based motion planning, our approach allows for significant performance improvements of algorithms such as PRM and RRT when planning for 3-D rigid bodies. We demonstrate the benefit of our approach when used in combination with RRT to construct a planning tree in an illustrative benchmark motion planning scenario.**

I. Introduction

The problem of 3-D rigid body motion planning arises in important application domains such as animation, physical simulation, manipulation and assembly planning, part removability analysis, path planning for free-flying robots, and analysis of molecular protein docking [16]. A 3-D rigid body has a six-dimensional configuration space (three degrees of freedom for position and three for orientation), for which sampling-based motion planners such as PRM [11] and RRT [17] have been shown to be particularly effective. The ability to check whether a straight-line connection between two configurations of a 3-D rigid body is collision-free with respect to obstacles in the environment is an essential ingredient of these algorithms. In fact, most of the computation time of PRM and RRT is spent on collision-checking such motions. Many implementations use a fixed-resolution approach, in which an evenly-spaced set of configurations along the motion are individually checked for collisions [4]. This method is simple to implement, yet does not guarantee that the motion is collision-free. In many applications, however, such a guarantee is important, which requires *continuous* or *exact* collision detection.

Approaches to exact collision detection can roughly be subdivided into four categories [22]: (i) methods based on solving algebraic equations [6], [7], [12], [20], (ii) methods based on the construction of (a conservative approximation of) the *swept volume* of the rigid body along its motion, which is then collision-checked against the obstacles [5], [10], [14], [22], [28], (iii) methods based on kinetic datastructures [1], [13], [15], and (iv) bounded-motion methods based on adaptive subdivision or conservative advancement [18], [19], [21], [24], [26], [27], [31], which rely on the ability of a collision-checker to efficiently compute (a conservative approximation of) the *distance* between workspace objects. Knowing the distance between the rigid body in some configuration and the workspace obstacles immediately determines the collision-status of surrounding configurations [2], [3], [25]. Namely, those configurations that are "closer" to the rigid body than the rigid body is to the obstacles are guaranteed to be collision-free (under the appropriate definition of "close"). The collision-status of an entire motion can then be determined exactly by computing the distance between the rigid body configured at the midpoint of the motion and the obstacles, and recursively repeating this procedure for the begin- and end-sections of the motion whose collision-status could not be determined from the computed distance [24].

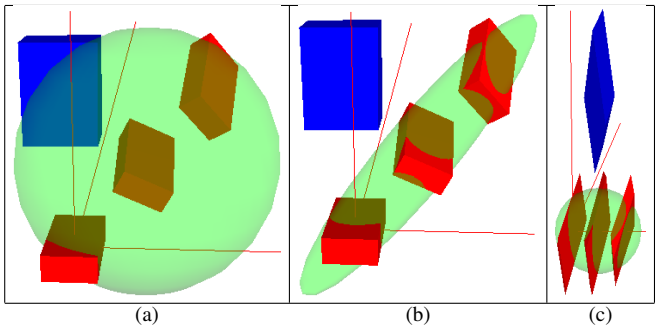

Fig. 1. Collision-checking the motion of a 3-D rigid body (red) among a workspace obstacle (blue). The beginpoint, midpoint, and endpoint of the rigid body motion are shown. Fig. (a) illustrates the traditional approach to exact collision-checking based on distance computations. The radius of the sphere (green) indicates the distance the obstacle needs to be away from the rigid body configured at the midpoint in order for the entire rigid body motion to be determined collision-free. Since the distance is much smaller, only a small portion of the motion can be determined collision-free with a single distance computation. It takes a total of 9 distance computations to guarantee that the entire rigid body motion is collision-free. Fig. (b) illustrates our new approach, in which the ellipsoid (green) indicates the *anisotropic* distance the obstacle needs to be away from the rigid body configured at the midpoint in order for the entire rigid body motion to be determined collision-free. Our method selects this ellipsoid such that its volume is *minimal*. By linearly transforming the workspace such that the ellipsoid becomes a sphere (Fig. (c)), one can use the Euclidean distance between the transformed rigid body at the midpoint and the transformed workspace obstacle. Our approach requires a total of only 3 distance computations to guarantee that this rigid body motion is collision-free.

Liang He and Jur van den Berg are with the School of Computing, University of Utah, Salt Lake City. E-mail: {liang, berg}@cs.utah.edu.

In this paper, we present a novel bounded-motion approach to exact collision detection for 3-D rigid body motions that is easy to implement and requires significantly fewer distance computations than existing methods. The key insight is that one can linearly transform the workspace, such that the distance between the transformed rigid body at the midpoint and the transformed obstacles determines the collision-status of a significantly larger portion of the rigid body motion. The insight lies in the information carried by a distance computation: it tells how far each point on the rigid-body is allowed to travel before it potentially collides with an obstacle (see Fig. 1(a)). However, for a given rigid-body motion, points on the rigid body tend to travel more in certain directions than in others (they travel parallel to the axis of translation plus some amount perpendicular to the axis of rotation). In fact, we will show that the motion of each point on the rigid body is contained within a oblique hourglass-shaped set. The minimum-volume enclosing ellipsoid of this hourglass thus defines an anisotropic distance function that provides more information about the amount of motion allowed along the dominant direction of travel (see Fig. 1(b)). By taking the ellipsoid with minimum volume, the likelihood that obstacles are contained within it is minimized, and hence larger portions of the motion can be determined collision-free with a single distance computation.

To compute anisotropic distances between workspace objects, the workspace is linearly transformed such that the ellipsoid becomes a sphere, and Euclidean distance computations are performed on the transformed workspace (see Fig. 1(c)). Our approach, as a result, can be used in combination with any collision-checker that is capable of performing both linear transformations and distance computations on object geometry. These operations are most efficient when the collision-checker is based on a broad phase that uses a type of bounding volume that is invariant to linear transformations, such as ellipsoids [23] or convex hulls [8]. Note that rectangular bounding boxes, such as AABB's [30] or OBB's [9] do not fall in this category. We used SOLID [29] in our implementation and experiments.

We implemented our approach within a Random Rapidly-exploring Tree (RRT) motion planning framework, and show using experiments in an illustrative benchmark scenario that our approach requires more than 20% fewer distance computations compared to using a traditional approach. The remainder of this paper is organized as follows. We formally define the problem we discuss in this paper in Section II. We present our approach in Section III and discuss experimental results in Section IV. We conclude in Section V.

II. Preliminaries and Problem Definition

A. Notation and Definitions

We use the following notational conventions in this paper. Vector sets $\mathcal{A}$ are denoted using calligraphics, vectors $\mathbf{a}$ are denoted using boldface, matrices A are denoted using upper-case italics, and scalars a are denoted in lower-case italics. Scalar and matrix multiplication and Minkowski addition of sets are defined as:

$$a\mathcal{X} = \{a\mathbf{x} \mid \mathbf{x} \in \mathcal{X}\}, \quad A\mathcal{X} = \{A\mathbf{x} \mid \mathbf{x} \in \mathcal{X}\}, \tag{1}$$

$$\mathcal{X} \oplus \mathcal{Y} = \{\mathbf{x} + \mathbf{y} \mid \mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}\}. \tag{2}$$

It follows that $\mathcal{X} \oplus \{\mathbf{x}\}$ denotes a translation of a set $\mathcal{X}$ by a vector $\mathbf{x}$. Also, multiplication by a matrix distributes over Minkowski addition and intersection of sets:

$$A(\mathcal{X} \oplus \mathcal{Y}) = A\mathcal{X} \oplus A\mathcal{Y}, \quad A(\mathcal{X} \cap \mathcal{Y}) = A\mathcal{X} \cap A\mathcal{Y}. \tag{3}$$

Further, we define $\mathcal{S}(d) \subset \mathbb{R}^3$ to denote a sphere of radius d centered at the origin:

$$\mathcal{S}(d) = \{\mathbf{q} \mid \|\mathbf{q}\| \leq d\}. \tag{4}$$

B. 3-D Rigid Body Motion

Let $\mathcal{B} \subset \mathbb{R}^3$ be the geometry of the rigid body defined within a local coordinate frame rigidly attached to the rigid body, and let $\mathcal{O}$ be the geometry of the workspace obstacles defined in a global coordinate frame. A *configuration* $\mathbf{x}$ of the rigid body is defined by a rotation matrix $R \in \mathbb{R}^{3\times3}$ from the local to the global coordinate frame and a position $\mathbf{p} \in \mathbb{R}^3$, such that the geometry of the rigid body configured at $\mathbf{x}$ in the global coordinate frame is given by $R\mathcal{B} \oplus \{\mathbf{p}\}$.

We consider the linearly interpolated motion of a 3-D rigid body from a configuration $\mathbf{x}_a$ as defined by R_a and $\mathbf{p}_a$ to a configuration $\mathbf{x}_b$ as defined by R_b and $\mathbf{p}_b$. Any such rigid body motion is associated with a translation along a straight-line between $\mathbf{p}_a$ and $\mathbf{p}_b$, and a rotation about a fixed axis $\hat{\mathbf{r}} = \mathbf{r}/\|\mathbf{r}\|$ by angle θ, where:

$$\mathbf{r} = \begin{bmatrix} (R_b R_a^T)_{3,2} - (R_b R_a^T)_{2,3} \\ (R_b R_a^T)_{1,3} - (R_b R_a^T)_{3,1} \\ (R_b R_a^T)_{2,1} - (R_b R_a^T)_{1,2} \end{bmatrix}, \tag{5}$$

$$\theta = \arctan2(\|\mathbf{r}\|, \mathrm{tr}(R_b R_a^T) - 1). \tag{6}$$

Note that $0 \leq \theta \leq \pi$.

Let the motion between the two configurations be parametrized by parameter $t \in [0, 1]$, such that $\mathbf{x}(0) = \mathbf{x}_a$ and $\mathbf{x}(1) = \mathbf{x}_b$. Let $\mathbf{p}(t)$ and $R(t)$ be the position vector and rotation matrix, respectively, of the rigid body during the motion, as given by (spherical) linear interpolation:

$$\mathbf{p}(t) = (1-t)\mathbf{p}_a + t\mathbf{p}_b, \quad R(t) = \exp(t\theta\lceil\hat{\mathbf{r}}\rceil)R_a, \tag{7}$$

where $\lceil\hat{\mathbf{r}}\rceil$ refers to the 3×3 skew-symmetric cross-product matrix of $\hat{\mathbf{r}}$. Finally, let $\mathcal{B}(t) \subset \mathbb{R}^3$ be the geometry of the rigid body in the global coordinate frame during the motion:

$$\mathcal{B}(t) = R(t)\mathcal{B} \oplus \{\mathbf{p}(t)\}. \tag{8}$$

C. Problem Definition

The problem we discuss in this paper is to determine whether the motion of the rigid body between a configuration $\mathbf{x}_a$ and a configuration $\mathbf{x}_b$ is collision-free with respect to the obstacles in the workspace, i.e. whether:

$$\forall\{t \in [0, 1]\} :: \mathcal{B}(t) \cap \mathcal{O} = \emptyset \; ? \tag{9}$$

For this, we assume we have access to a collision-checker that is capable of performing linear transformations and

distance computations on 3-D geometry. More formally, for given sets $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^3$, matrices $A, B \in \mathbb{R}^{3\times3}$, and vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^3$ it can compute:

$$d(A\mathcal{X} \oplus \{\mathbf{a}\}, B\mathcal{Y} \oplus \{\mathbf{b}\}) = \min\{\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}\} \|(A\mathbf{x} + \mathbf{a}) - (B\mathbf{y} + \mathbf{b})\|. \quad (10)$$

Note that this function returns zero if $(A\mathcal{X} \oplus \{\mathbf{a}\})$ and $(B\mathcal{Y} \oplus \{\mathbf{b}\})$ intersect (i.e. collide), and that for any $\delta \geq 0$:

$$d(\mathcal{X}, \mathcal{Y}) > \delta \iff (\mathcal{X} \oplus \mathcal{S}(\delta)) \cap \mathcal{Y} = \emptyset. \quad (11)$$

The goal is to answer Eq. (9) with as few distance computations as possible.

We also assume we are given the *radius* r of the rigid body, defined as the distance between the origin (of the local coordinate frame) and the farthest point on the geometry of the rigid body:

$$r = \max\{\mathbf{q} \in \mathcal{B}\} \|\mathbf{q}\|. \quad (12)$$

III. Exact Collision-Checking

A. The Rigid Body Motion Hourglass

For simplicity, let us assume, *without loss of generality*, that the axis of rotation is aligned with the $\mathbf{z}$-axis of the global coordinate frame (one can always rotate the entire scene such that this holds, and we'll make this explicit in Section III-D). Then, let $(\Delta x, \Delta y, \Delta z)^T = \mathbf{p}(1) - \mathbf{p}(0)$, i.e. the change in position between the beginpoint and endpoint of the rigid body motion.

Let $\mathbf{b} \in \mathcal{B}$ be any point on the rigid body, and let $\mathbf{b}(t) = R(t)\mathbf{b} \oplus \{\mathbf{p}(t)\}$ be its position in the global coordinate frame at point t along the rigid body motion. Then as point $\mathbf{b}$ travels from $\mathbf{b}(t_0)$ to $\mathbf{b}(t)$ (for $0 \leq t_0, t \leq 1$), it moves by $(t - t_0)[\Delta x, \Delta y, \Delta z]^T$ as a result of the translation of the rigid body, and *at most* an additional distance of $|(t - t_0)r\theta|$ in the $\mathbf{xy}$-plane as a result of the rotation of the rigid body (it cannot move in the $\mathbf{z}$-direction as a result of rotation, as the rotation axis is aligned with the $\mathbf{z}$-axis). Hence, we have:

$$\mathbf{b}(t) \in \{\mathbf{b}(t_0) + (t - t_0)[\Delta x, \Delta y, \Delta z]^T + [x, y, 0]^T \mid x^2 + y^2 \leq ((t - t_0)r\theta)^2\}. \quad (13)$$

Consequently, for all $t \in [t_0 - \tau, t_0 + \tau]$, the point $\mathbf{b}(t)$ is contained within an oblique circular double cone $\mathcal{H}(\tau)$, which we call the *rigid body motion hourglass*, with its apex translated to $\mathbf{b}(t_0)$ (see Fig. 2):

$$\forall\{t \in [t_0 - \tau, t_0 + \tau]\} :: \mathbf{b}(t) \in \{\mathbf{b}(t_0)\} \oplus \mathcal{H}(\tau), \quad (14)$$

where

$$\mathcal{H}(\tau) = \bigcup_{t\in[-\tau,\tau]} \left\{ t \begin{bmatrix}\Delta x \\ \Delta y \\ \Delta z\end{bmatrix} + \begin{bmatrix}x \\ y \\ 0\end{bmatrix} \;\middle|\; x^2 + y^2 \leq (tr\theta)^2 \right\}. \quad (15)$$

Since Eq. (14) holds for all points $\mathbf{b} \in \mathcal{B}$ on the rigid body, we have:

$$\forall\{t \in [t_0 - \tau, t_0 + \tau]\} :: \mathcal{B}(t) \subset \mathcal{B}(t_0) \oplus \mathcal{H}(\tau). \quad (16)$$

This observation lies at the basis of exact collision-checking based on distance computations.

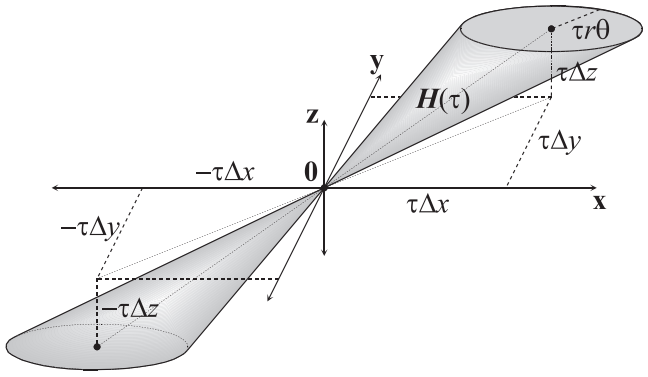

Fig. 2. The rigid body motion hourglass $\mathcal{H}(\tau)$ as defined in Eq. (15) contains all possible displacements of any point on the rigid body $\mathcal{B}(t_0)$ over the portion $[t_0 - \tau, t_0 + \tau]$ of the rigid body motion.

B. Distance Computations for Exact Collision-Checking

The radius $d_{\min}(\tau)$ of the smallest sphere circumscribing $\mathcal{H}(\tau)$ is:

$$d_{\min}(\tau) = \tau\sqrt{(\sqrt{\Delta x^2 + \Delta y^2} + r\theta)^2 + \Delta z^2}. \quad (17)$$

Hence, if the distance between the rigid body geometry $\mathcal{B}(t_0)$ and the obstacle geometry $\mathcal{O}$ is larger than $d_{\min}(\tau)$, the portion of the rigid body motion for $t \in [t_0 - \tau, t_0 + \tau]$ is collision-free:

$$\begin{aligned} & d(\mathcal{B}(t_0), \mathcal{O}) > d_{\min}(\tau) \\ \overset{\text{(i)}}{\iff} & (\mathcal{B}(t_0) \oplus \mathcal{S}(d_{\min}(\tau))) \cap \mathcal{O} = \emptyset \\ \overset{\text{(ii)}}{\implies} & (\mathcal{B}(t_0) \oplus \mathcal{H}(\tau)) \cap \mathcal{O} = \emptyset \\ \overset{\text{(iii)}}{\implies} & \forall\{t \in [t_0 - \tau, t_0 + \tau]\} :: \mathcal{B}(t) \cap \mathcal{O} = \emptyset, \end{aligned} \quad (18)$$

where equivalence (i) follows from Eq. (11), implication (ii) follows from the fact that $\mathcal{H}(\tau) \subset \mathcal{S}(d_{\min}(\tau))$, and implication (iii) follows from Eq. (16). Hence, the portion $[t_0 - \tau, t_0 + \tau]$ of the rigid body motion that is guaranteed to be collision-free given $d(\mathcal{B}(t_0), \mathcal{O})$ is found by solving $d(\mathcal{B}(t_0), \mathcal{O}) = d_{\min}(\tau)$ for τ:

$$\tau = d(\mathcal{B}(t_0), \mathcal{O}) / \sqrt{(\sqrt{\Delta x^2 + \Delta y^2} + r\theta)^2 + \Delta z^2}. \quad (19)$$

The above analysis lies at the basis of existing exact collision-checking approaches based on distance computations such as [19], [24], [26], [27], [31].

C. Efficient Exact Collision-Checking

The downside of the above approach is that in general the enclosing sphere $\mathcal{S}(d_{\min}(\tau))$ has a large volume compared to the hourglass $\mathcal{H}(\tau)$, resulting in a very conservative collision-check that would in many cases only determine the collision-status of a small portion of the rigid body motion. In fact, the ratio of the volume of the enclosing sphere and the volume of the hourglass is *unbounded* (see Fig. 4). To improve this, we propose the following approach, in which we linearly *transform* the workspace (and with it the hourglass) by a transformation T that *minimizes* the ratio of the volume of the enclosing sphere of the transformed hourglass $T\mathcal{H}(\tau)$ and the volume of the transformed hourglass $T\mathcal{H}(\tau)$. We can

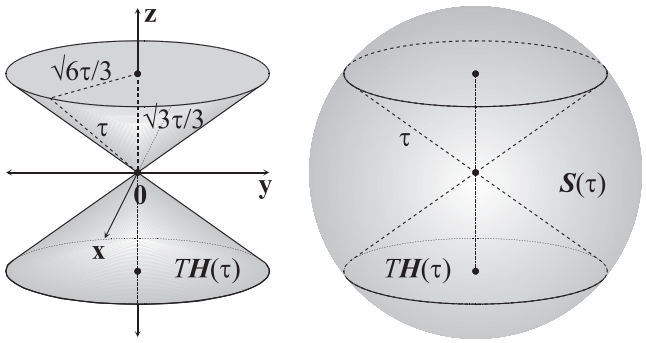

Fig. 3. Left: the transformation $T = T_2T_1$ transforms the hourglass $\mathcal{H}(\tau)$ such that it is no longer oblique and has height $\sqrt{3}\tau/3$ and base radius $\sqrt{6}\tau/3$. Right: the sphere $\mathcal{S}(\tau)$ with radius τ is the smallest enclosing sphere of the transformed hourglass $T\mathcal{H}(\tau)$. The volume of $\mathcal{S}(\tau)$ is $3\sqrt{3}$ times the volume of $T\mathcal{H}(\tau)$.

then use the distance computation of the collision-checker on the transformed workspace.

We proceed as follows. First, we perform a (volume-preserving) *shear*-transformation T_1 that lets the axis of the hourglass $T_1\mathcal{H}(\tau)$ be aligned with the **z**-axis, such that it is no longer oblique:

$$T_1 = \begin{bmatrix} 1 & 0 & -\Delta x/\Delta z \\ 0 & 1 & -\Delta y/\Delta z \\ 0 & 0 & 1 \end{bmatrix}. \tag{20}$$

Then we perform a scaling T_2, such that the transformed hourglass $T_2T_1\mathcal{H}(\tau)$ has a height of $\sqrt{3}\tau/3$ and a base radius of $\sqrt{6}\tau/3$, and fits within a sphere of radius τ (see Fig. 3):

$$T_2 = \begin{bmatrix} \sqrt{6}/(3r\theta) & 0 & 0 \\ 0 & \sqrt{6}/(3r\theta) & 0 \\ 0 & 0 & \sqrt{3}/(3\Delta z) \end{bmatrix}. \tag{21}$$

Let $T = T_2T_1$ denote the combined transformation.

The ratio of the volume of the enclosing sphere $\mathcal{S}(\tau)$ and the volume of the transformed hourglass $T\mathcal{H}(\tau)$ can be shown to be $3\sqrt{3} \approx 5.196$ (see Fig. 3). In fact, the scaling T_2 was chosen such that this value of $3\sqrt{3}$ is *minimal*. Hence, "untransforming" the sphere $\mathcal{S}(\tau)$ gives the *minimum-volume enclosing ellipsoid* $T^{-1}\mathcal{S}(\tau)$ of the original hourglass $\mathcal{H}(\tau)$ with the same volume ratio of $3\sqrt{3}$ (see Fig. 4). In contrast the ratio of the volume of the minimum enclosing sphere $\mathcal{S}(d_{\min}(\tau))$ of $\mathcal{H}(\tau)$ and the volume of $\mathcal{H}(\tau)$ is *unbounded* (see Fig. 4). In this light, the ellipsoid $T^{-1}\mathcal{S}(\tau)$ can be interpreted as defining an optimal *anisotropic* distance function for collision-checking the given rigid body motion.

Therefore, when using the linear transformation T on the workspace, it is likely that a significantly larger portion of the rigid body motion can be determined collision-free with a single distance computation of the collision-checker. If the distance between the transformed rigid body geometry $T\mathcal{B}(t_0)$ and the transformed obstacle geometry $T\mathcal{O}$ is larger than τ, the portion of the rigid body motion for $t \in [t_0 - \tau, t_0 + \tau]$ is guaranteed to be collision-free:

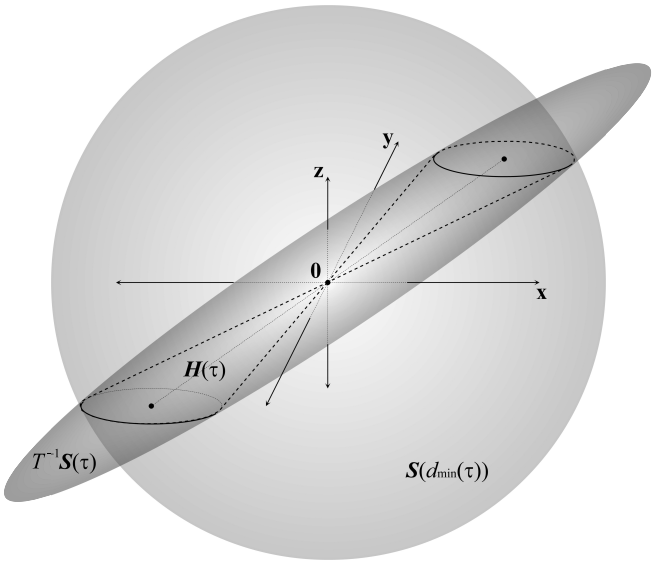

Fig. 4. The hourglass $\mathcal{H}(\tau)$, its smallest enclosing sphere $\mathcal{S}(d_{\min}(\tau))$, and its minimum-volume enclosing ellipsoid $T^{-1}\mathcal{S}(\tau)$. The ratio of the volume of the enclosing ellipsoid $T^{-1}\mathcal{S}(\tau)$ and the volume of the hourglass $\mathcal{H}(\tau)$ is $3\sqrt{3}$. In contrast, the ratio of the volume of the enclosing sphere $\mathcal{S}(d_{\min}(\tau))$ and the hourglass $\mathcal{H}(\tau)$ is unbounded.

$$\begin{aligned} & d(T\mathcal{B}(t_0), T\mathcal{O}) > \tau \\ \overset{\text{(i)}}{\iff} & (T\mathcal{B}(t_0) \oplus \mathcal{S}(\tau)) \cap T\mathcal{O} = \emptyset, \\ \overset{\text{(ii)}}{\implies} & (T\mathcal{B}(t_0) \oplus T\mathcal{H}(\tau)) \cap T\mathcal{O} = \emptyset, \\ \overset{\text{(iii)}}{\iff} & (\mathcal{B}(t_0) \oplus \mathcal{H}(\tau)) \cap \mathcal{O} = \emptyset, \\ \overset{\text{(iv)}}{\implies} & \forall\{t \in [t_0 - \tau, t_0 + \tau]\} :: \mathcal{B}(t) \cap \mathcal{O} = \emptyset, \end{aligned} \tag{22}$$

where equivalence (i) follows from Eq. (11), implication (ii) follows from the fact that $T\mathcal{H}(\tau) \subset \mathcal{S}(\tau)$, equivalence (iii) follows from multiplying both sides by T^{-1} and applying Eq. (3), and implication (iv) follows from Eq. (16).

We apply this insight in a recursive collision-checking algorithm (see Fig. 5). Suppose we want to check whether the portion $[t_1, t_2]$ of the rigid body motion is collision-free (to check the entire rigid body motion, set $t_1 = 0$ and $t_2 = 1$). Then, we compute the distance $\tau = d(T\mathcal{B}(t_{\text{mid}}), T\mathcal{O})$ between the transformed rigid body $T\mathcal{B}(t_{\text{mid}})$ at the midpoint $t_{\text{mid}} = (t_1 + t_2)/2$ of the motion and the transformed obstacles $T\mathcal{O}$. If $\tau > (t_2 - t_1)/2$, the entire portion of the rigid body motion is collision-free, and if $\tau = 0$ (i.e. $\mathcal{B}(t_{\text{mid}})$ collides with $\mathcal{O}$), the rigid body motion is not collision-free. For other values of τ, only the portion $[t_{\text{mid}} - \tau, t_{\text{mid}} + \tau]$ of the rigid body motion can be determined collision-free, and we recursively apply the algorithm to check whether the remaining portions $[t_1, t_{\text{mid}} - \tau]$ and $[t_{\text{mid}} + \tau, t_2]$ are collision-free as well. We suggest that the recursion is evaluated in a breadth-first order, such that the distance computations are distributed more evenly across the rigid body motion, increasing the likelihood that a potential collision is encountered early.

We note that the transformation T is degenerate when $\theta = 0$ or $\Delta z = 0$. In these cases, a "small" value can safely

CHECK(t_1, t_2)

1: $t_{\text{mid}} = (t_1 + t_2)/2$.
2: $\tau = d(T\mathcal{B}(t_{\text{mid}}), T\mathcal{O})$.
3: **if** $\tau > (t_2 - t_1)/2$ **then**
4: **return true**.
5: **else if** $\tau = 0$ **then**
6: **return false**.
7: **else**
8: **return** CHECK$(t_1, t_{\text{mid}} - \tau) \wedge$ CHECK$(t_{\text{mid}} + \tau, t_2)$.

Fig. 5. A recursive algorithm for collision-checking a portion $[t_1, t_2]$ of the rigid body motion. The algorithm returns "true" if the portion $[t_1, t_2]$ of the rigid-body motion is collision-free and "false" if it is not collision-free. Calling this algorithm for $t_1 = 0$ and $t_2 = 1$ checks the collision-status of the entire rigid-body motion.

be used instead. It indeed appears that for the seemingly "simpler" problems of 2-D rigid body motion ($\Delta z = 0$) and 3-D translational motion ($\theta = 0$), our approach does not have straightforward analogies.

D. General Axes of Rotation

In the above, we made the assumption that the axis of rotation is aligned with the $\mathbf{z}$-axis of the global coordinate frame. This is in general not the case, but we can rotate the entire scene by T_0 such that this holds. This rotation T_0 is given by:

$$T_0 = \exp(\lceil \hat{\mathbf{r}} \times \mathbf{z} \rceil \arctan2(\|\hat{\mathbf{r}} \times \mathbf{z}\|, \hat{\mathbf{r}} \cdot \mathbf{z})/\|\hat{\mathbf{r}} \times \mathbf{z}\|), \quad (23)$$

where $\mathbf{z} = [0, 0, 1]^T$. We can then use the method as described above by defining $[\Delta x, \Delta y, \Delta z]^T = T_0(\mathbf{p}(1) - \mathbf{p}(0))$ and using $T = T_2 T_1 T_0$. The rest remains exactly the same.

IV. Experimental Results

We implemented our approach as part of a simple RRT algorithm [17], and experimented with it in the scenario of Fig. 6, which measures 100 units of length in each dimension. The obstacles consist of 1330 tetrahedra generated by randomly sampling four points on spheres with radii of 5 units of length whose centers lie on a grid with spacing 10. The rigid body is a hook with two legs each of length 5.

We built an RRT-tree in this environment by continually sampling a uniform-random configuration [16] from the domain and attempt to connect it to the nearest node in the tree (we used the Euclidean distance between the positions of the configurations as the distance metric). If the connection succeeds, i.e. the rigid-body motion between the two configurations is collision-free, the sampled configuration is added as a node to the tree. If the connection fails, the sampled configuration is discarded. We repeated this until the tree consisted of 10,000 nodes. The root of the tree corresponds to a rigid-body configuration in the center of the domain.

We performed the above experiment with two methods for collision-checking connections between nodes: our approach, and a traditional approach similar to [24] based on distance computations without linear transformations of the workspace (see Section III-B). For both approaches the distance computations were performed using SOLID [29]. In both experiments we used the same sequence of random numbers, so in both cases the same connections are collision-checked. The results are shown in Table IV and in Fig. 7.

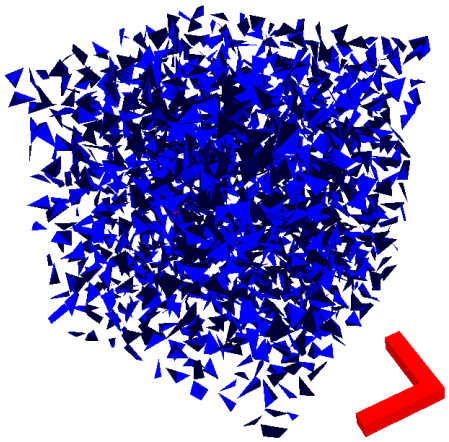

Fig. 6. The motion planning scenario in which we evaluated our approach. The obstacles are 1330 randomly generated tetrahedra (blue) over a volume of 100 units of length in each dimension, and the rigid body (red, not shown to scale) is a hook with legs of 5 units of length.

Constructing a tree containing 10,000 nodes required a total of 19,809 connections to be collision-checked. Of these, 9,999 connections were successful (collision-free) while 9,810 connections failed. In total, our approach required 23.9% fewer distance computations to collision-check all connections than the standard approach (89,155 vs. 116,132 distance computations). Interestingly, if we look only to the 9,999 successful connections, we see that our approach performed even better compared to the traditional approach: 59,321 vs. 83,787 distance computations, an improvement of 29.2%. For the 9,810 failed connections on the other hand, the difference between our approach and the traditional approach was less significant: 29,834 vs. 32,345 distance computations, an improvement of 7.7%. This is because it is easier to determine a connection unsuccessful, as the recursive algorithm (Fig. 5) can be terminated as soon as a collision is encountered. Indeed, even as the number of successful connections was about the same as the number of failed connections in this experiment, for both methods the majority of the distance computations were spent on determining connections to be collision-free. Our approach required an average of 5.9 distance computations to determine a connection collision-free, and only 3.0 to determine a connection unsuccessful (versus 8.4 and 3.3 distance computations, respectively, for the traditional approach). As a result, the improvement our method shows for successful connections carries a large weight in the improvement overall.

As the RRT tree grows and covers larger portions of the free configuration space, the length of the connections that are collision-checked decreases (see Fig. 7). Somewhat surprisingly, this seems not to have an effect on the average number of distance computations required to collision-check a connection: both methods show a fairly linear growth of the cumulative number of distance computations as a function of the size of the tree.

V. Conclusion

We have presented a new method for exact collision-checking of linear motions of 3-D rigid bodies that can

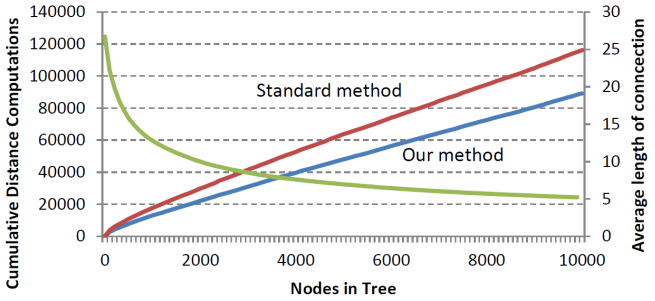

Fig. 7. Results of our implementation. Shown is the cumulative number of distance computations (left axis) required for our method (blue line) and the traditional standard method (red line) as the number of nodes in the tree increases. The green line indicates the length of the connections that are collision-checked as the tree grows, averaged over a sliding window of 100 connections (right axis).

TABLE I

NUMBER OF DISTANCE COMPUTATIONS (10,000 NODES)

Connections (num.)	std. method (av.)	our method (av.)	improv.
All (19,809)	116,132 (5.9)	89,155 (4.5)	23.2%
Successful (9,999)	83,787 (8.4)	59,321 (5.9)	29.2%
Failed (9,810)	32,345 (3.3)	29,834 (3.0)	7.7%

be used in combination with any collision-checker that is capable of performing linear transformations and distance computations on 3-D geometry. Our approach is based on the observation that by performing a carefully selected *linear transformation* of the workspace a single distance computation carries information about the collision status of larger portions of rigid-body motions. Our experiments show that our approach requires on average at least 20% fewer distance computations than traditional approaches.

There are a number of avenues to further improve our approach that we would like to explore in future work. The first concerns the radius of the rigid body, which we defined as the radius of its smallest enclosing sphere. In fact, however, the rigid body has a different, smaller radius with respect to each rotation axis. Since the radius has a quadratic effect on the volume of the *rigid body motion hourglass*, computing this smaller radius may be worth the effort. A second potentially interesting observation is that the volume of this hourglass may serve as an ideal *distance metric* [16] for PRMs and RRTs. This would mean that these algorithms would attempt those connections with the highest a priori likelihood of being collision-free first, assuming that indeed the volume of the hourglass is proportional to this likelihood. A theoretical underpinning of this conjecture is subject of ongoing work.

REFERENCES

[1] P. Agarwal, J. Basch, L. Guibas, J. Hershberger, L. Zhang. Deformable free space tilings for kinetic collision detection. *Int. Journal of Robotics Research* 21(3):179-197, 2002.

[2] J. Bialkowski, S. Karaman, M. Otte, E. Frazzoli. Efficient collision checking in sampling-based motion planning. *Proc. Workshop on Algorithmic Foundations of Robotics*, 2012.

[3] O. Brock, L. Kavraki. Decomposition-based motion planning: a framework for real-time motion planning in high-dimensional configuration spaces. *Proc. IEEE Int. Conf. on Robotics and Automation*, 2001.

[4] S. Cameron. A study of the clash detection problem in robotics. *Proc. IEEE Int. Conf. on Robotics and Automation*, 1985.

[5] M. Campen, L. Kobbelt. Polygonal boundary evaluation of Minkowski sums and swept volumes. *Computer Graphics Forum* 29(5):1613-1622, 2010.

[6] J. Canny. Collision detection for moving polyhedra. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 8(2):200-209, 1986.

[7] Y.-K. Choi, W. Wang, Y. Liu, M.-S. Kim. Continuous collision detection for two moving elliptic disks. *IEEE Trans. on Robotics* 22(2):213-224, 2006.

[8] S. Ehmann, M. Lin. Accurate and fast proximity queries between polyhedra using convex surface decomposition. *Computer Graphics Forum* 20(3):500-511, 2001.

[9] S. Gottschalk, M. Lin, D. Manocha. OBBTree: a hierarchical structure for rapid interference detection. *Proc. ACM Conf. on Computer Graphics and Interactive Techniques*, 1996.

[10] J. Himmelstein, E. Ferre, J.-P. Laumond. Swept volume approximation of polygon soups. *IEEE Trans. Automation Science and Engineering* 7(1):177-183, 2010.

[11] L. Kavraki, P. Svestka, J.-C. Latombe, M. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. on Robotics and Automation* 12(4):566-580, 1996.

[12] B. Kim, J. Rossignac. Collision prediction for polyhedra under screw motions. *Proc. ACM Conf. on Solid Modeling and Applications*, 2003.

[13] D. Kim, L. Guibas, S. Shin. Fast collision detection among multiple moving spheres. *IEEE Trans. on Visualization and Computer Graphics* 4(3):230-242, 1998.

[14] Y. Kim, G. Varadhan, M. Lin, D. Manocha. Fast swept volume approximation of complex polyhedral models. *Computer-Aided Design* 36(11):1013-1027, 2004.

[15] D. Kirkpatrick, J. Snoeyink, B. Speckmann. Kinetic collision detection for simple polygons. *Proc. ACM Symp. on Computational Geometry*, 2000.

[16] J. Kuffner. Effective sampling and distance metrics for 3d rigid body path planning. *IEEE Int. Conf. on Robotics and Automation*, 2004.

[17] J. Kuffner, S. LaValle. RRT-connect: an efficient approach to single-query path planning. *IEEE Int. Conf. on Robotics and Automation*, 2000.

[18] B. Mirtich. *Impulse-based dynamic simulation of rigid body systems*. PhD. dissertation, University of California, Berkeley, 1996.

[19] J. Pan, L. Zhang, D. Manocha. Collision-free and smooth trajectory computation in cluttered environments. *Int. Journal of Robotics Research* 31(10):1155-1175, 2012.

[20] S. Redon, A. Kheddar, S. Coquillart. An algebraic solution to the problem of collision detection for rigid polyhedral objects. *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000.

[21] S. Redon, A. Kheddar, and S. Coquillart. Fast continuous collision detection between rigid bodies. *Computer Graphics Forum* 21(3):279-287, 2002.

[22] S. Redon, Y. Kim, M. Lin, D. Manocha. Fast continuous collision detection for articulated models. *Proc. ACM Symp. on Solid Modeling and Applications*, 2004.

[23] E. Rimon, S. Boyd. Obstacle collision detection using best ellipsoid fit. *Journal of Intelligent and Robotic Systems* 18(2):105-126, 1997.

[24] F. Schwarzer, M. Saha, J.-C. Latombe. Exact collision checking of robot paths. *Workshop on Algorithmic Foundations of Robotics*, 2002.

[25] A. Shkolnik, R. Tedrake. Sample-based planning with volumes in configuration space. *arXiv*:1109.3145, 2011.

[26] M. Tang, Y. Kim, D. Manocha. C2A: Controlled conservative advancement for continuous collision detection of polygonal models. *Proc. IEEE Int. Conf. on Robotics and Automation*, 2009.

[27] M. Tang, Y. Kim, D. Manocha. CCQ: Efficient local planning using connection collision query. *Proc. Workshop on Algorithmic Foundations of Robotics*, 2010.

[28] H. Täubig, B. Bäuml, U. Frese. Real-time swept volume and distance computation for self collision detection. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2011.

[29] G. van den Bergen. *Collision detection in interactive 3d environments*. Morgan Kaufmann Publishers, 2004.

[30] G. van den Bergen. Efficient collision detection of complex deformable models using AABB trees. *Journal of Graphics Tools* 2(4):1-13, 1997.

[31] X. Zhang, M. Lee, Y. Kim. Interactive continuous collision detection for non-convex polyhedra. *The Visual Computer* 22(9-11):749-760, 2006.