Nearly Uniform Sampling on Surfaces with Applications to Motion Planning

Mukulika Ghosh
mghosh@cse.tamu.edu

Hsin-Yi (Cindy) Yeh
hyeh@cse.tamu.edu

Shawna Thomas
sthomas@cse.tamu.edu

Nancy M. Amato
amato@cse.tamu.edu

Technical Report TR13-005

Parasol Lab.
Department of Computer Science and Engineering
Texas A&M University

April 08, 2013

Abstract

Uniform sampling on/near the surfaces of complex geometric objects has application in mesh generation, remodeling, motion planning, and others. In short, it provides an alternate and approximate way to represent the object which for some applications can be used in place of the original object, either to save computation time or to remove detail that is not needed, and perhaps even distracting. For example, in motion planning, high clearance paths can be safer and in some cases computed more efficiently by ignoring concavities on obstacle surfaces. However, generating samples that uniformly cover a surface becomes more difficult and costly as the model complexity increases. In this work, we present a two phase approach to generate points that "nearly" uniformly cover model surfaces by first decomposing the model into approximately convex pieces and then uniformly sampling on the convex hull surfaces of these pieces. We implement the first phase using Fast Approximate Convex Decomposition (FACD), a method that efficiently partitions the model into approximately convex pieces such that their concavity does not exceed a specified value τ; we call the convex hulls of the approximately convex pieces τ-surfaces. Then, the Uniform Obstacle-Based Probabilistic Roadmap Method (UOBPRM) is used to generate points distributed uniformly on the τ-surfaces by finding intersections between uniformly sampled fixed length line segments and the τ-surfaces. Operating on the τ-surfaces instead of the the original surfaces is expected to be more efficient since the convex hulls have reduced complexity over the original input model. We evaluate our method with respect to time and error in average distance to the original input model and show that this scheme produces increased efficiency in time with reduced distance error as compared to using a convex hull approximation of the entire model and other decomposition approaches. We also show that our sampling strategy can be used in a sampling-based motion planning method to solve a challenging motion planning problem more efficiently than previous methods.

1 Introduction

Sampling near the surface of polyhedral objects helps in understanding the underlying topology and geometric features of large complex models. Point cloud covering has applications in mesh regeneration [13] and matching [28], rendering [1, 16], and others [4]. Generating a uniform distribution around an object is a popular problem as it gives even coverage over the surface area of the input model. Poisson disk sampling methods generate almost uniform samples on the model surface [12]. Offset methods generate samples at a fixed resolution from the model surface [9, 10].

Sample generation is also a key concept in motion planning. The motion planning problem is stated as finding a valid path for a moveable object from a start configuration to a goal configuration. Exact solutions are computationally infeasible as the degree of freedom of the moveable object increases [5]. Thus, attention has turned to randomized approaches. One highly successful class of algorithms is the Probabilistic Roadmap Method (PRM) [19, 20, 29]. PRM samples points in the configuration space or C-space (the set of all moveable object placements, valid or not), retains valid ones, and then connects neighboring points with valid paths to construct a motion graph, or roadmap. Basic graph search techniques are applied on this roadmap to extract the final trajectory from the start to the goal. Many PRM variants exist to address various issues like maximum path clearance [32] and difficulty in sampling narrow passages [6, 17]. Of particular note are the various obstacle-based methods [3, 2, 6, 17] that bias sampling towards regions near the surfaces of C-space as increased sampling in these areas assists in building roadmaps whose connectivity reflects the underlying connectivity of the free C-space. While the OBPRM method [3, 2] was observed to work well in practice, it could offer no guarantee regarding the distribution of the sampled points on the C-obstacle surfaces and indeed the distribution was highly dependent on the obstacle's shape. Recently, a new method, Uniform Obstacle-Based PRM (UOBPRM) [33] was proposed that is guaranteed to generate points uniformly distributed on the C-space surfaces. While more efficient than Gaussian [6] and Bridge-test [17] sampling, its running time is correlated with the complexity of the obstacle models since it works by finding the intersections of a set of uniformly distributed fixed length line segments with the obstacle surfaces.

Generating a uniform sample distribution around an input surface, whether a polyhedral model or a set of obstacles in a motion planning application, becomes increasingly difficult as the complexity of the input surface increases. In addition, for many applications, a "nearly" uniform distribution can be tolerated. In this paper, we present a two phase approach to generate "nearly" uniformly distributed samples near an input surface by first decomposing the input into approximately convex pieces and then uniformly sampling around the convex hulls of these pieces. We implement the first phase using Fast Approximate Convex Decomposition (FACD) [15, 14], a method that efficiently partitions the model into approximately convex pieces such that their concavity does not exceed a specified value τ. It is desirable here to perform an approximate decomposition because computing the exact convex decomposition of large complex models typically yields a large number of small components. Samples are generated on the convex hulls of these decomposed pieces using UOBPRM. Thus, using UOBPRM on the convex hulls of FACD components, called τ-surfaces, produces a uniform distribution over the τ-surfaces which results in a "nearly" uniform distribution over the original input model. In addition, the efficiency of UOBPRM is greatly increased over the original input model. The error in uniformity can be fairly traded with the efficiency of the sampling method as long as structural concavities are not ignored.

We validate the efficiency of our method by generating samples using UOBPRM on the original input model, on its convex hull, and on the collection of convex hulls of the decomposed parts. We compared time and distance error due to approximation. We also analyze the advantage of using FACD by comparing to seven other decompositions provided in Princeton Benchmark [8]. Finally, we applied our method to solve a motion planning problem.

2 Related Work

Sampling around Polyhedral Objects. Generating uniform point samples around a polyhedral object is studied extensively for its application in re-triangulation, object representation using a point data set, and other surface properties. Poisson disk sampling, or *darting*, creates uniform samples such that the geodesic distance between two samples is at least some threshold distance. Turk describes a method of uniformly sampling by distributing samples on the surface using Lloyd's relaxation algorithm [26] and rearranging them until they are uniformly distributed [30]. Cline et. al. describes an "optimized" darting algorithm by dividing the surface into fragments and using the fragments to ensure no two darts are too close to each other [11]. Nehab and Shilane describe a stratified point distribution method using subdivision of the area surrounding the mesh by octrees and placing the samples at each octree node [27]. Cao et. al. introduces a method of mesh simplification based on point sampling around the mesh [7]. They perform two level of sampling: *global sampling* gives an uniform distribution of points around the object and *local sampling* is performed on decomposed patches of the model to preserve the local details in the sampled points.

Sampling in Motion Planning. Probabilistic Roadmap Methods (PRM) [19, 20, 29] first sample the free C-space to generate the nodes of the motion graph, or roadmap. Thus, sampling is one of the key components to the method's ability to model the topology of the free C-space, particularly in narrow passages which lie near C-obstacle surfaces. As such, there have been many algorithms aimed at sampling these narrow passages near C-obstacle surfaces. For example, Gaussian sampling [6] looks for pairs of samples a Gaussian distance d apart such that one is valid and one is invalid. Bridge-test sampling [17] is similar in that it looks for pairs of invalid samples a

Gaussian distance d apart such that their midpoint is valid. Obstacle-Based PRM (OBPRM) [3, 2] samples nodes near the obstacle surface by performing binary searches along random rays initiating from inside the obstacle. However, there is no guarantee on the sample distribution which depends heavily on the obstacle's shape. Uniform Obstacle-Based PRM [33] samples uniformly distributed configurations around the obstacles by finding intersections between uniformly sampled fixed length line segments and the obstacle's surface.

Decomposition in Motion Planning. Approximation of polyhedral objects into convex shapes for collision detection is a common practice used in motion planning algorithms. For example, Hubbard approximates polyhedral obstacles into spheres [18]. Although spheres allow for faster collision detection queries, it works better for polyhedra composed of spherical volumes like a teddy model than for other obstacles like a table or chair model which can be better approximated using bounding box. Liu et. al. proposes a new segmentation algorithm to aid collision detection by decomposing the surface and using the convex hulls of the decomposed convex patch for collision detection [25]. Collision detection is faster using convex patch segmentation but yields a large number of components as compared to volumetric segmentation.

Decomposition has also been used to handle the narrow or difficult passage problem. Berg and Overmars decompose the free workspace into cells and label them based on their properties to guide sampling through narrow passages [31]. Lien describes approximate convex decomposition of the free space by biasing sampling towards the decomposition boundaries and the centroid of the decomposed components to address the "narrow passage" issue [21]. In [24], Lien and Lu use approximate star-shaped decomposition of the point set representing the contact surface to solve both deterministic and probabilistic motion planing problems. All of these methods decompose the free space or its boundary with the obstacle space to solve the motion planning problem.

3 Our Approach

A "nearly" uniform sampling creates samples, or points, around a model or a surface such that the distribution is uniform with respect to an approximated representation of the model or the surface. In nearly-uniform sampling, the uniformity in point distribution is traded for the efficiency of the sampling method. Generally, large complex models have details such as surface texture or minor undulations which can be ignored while preserving important structural concavities of the model. These surface details increase the surface area of the model and hence adversely affect sampling time as more samples are required to get better model coverage in order to encode such details. For example, spending time sampling in such concave regions usually does not provide much useful information when solving a motion planning problem. Using an approximation of the model to generate samples not only saves the number of samples needed but also expedites the sampling process while still providing enough information about the model. Hence, using an already proven uniform sampling method on the approximated model renders points that are uniform to the approximated surface but nearly-uniform to the original surface. This type of distribution is desirable for noisy models and approximate model coverage.

In this paper, the model is approximated by the convex hulls of its approximate convex decomposition and uniform samples are generated on this approximated model. Algorithm 1 gives the basic outline of the method. Given a set of input models, they are first decomposed into approximately convex pieces such that the maximum concavity of each piece does not exceed a user-defined tolerance value τ. The convex hulls of these decomposed pieces are computed and are known as *τ-surfaces or volumes*. This constitutes steps 1–6 in Algorithm 1. These τ-surfaces are then uniformly sampled to generate the set of "nearly" uniform points around the model. Any point internal to any of the convex hulls of the model decomposition is discarded. Steps 8–13 in Algorithm 1 provide the pseudocode.

Algorithm 1 Nearly Uniform Sampling over Model Surfaces(M)

Input: A set of models M.
Output: A set of points P around M.

1: **for** each $m_i \in M$ **do**
2: $\quad D_i \leftarrow$ Decompose m_i into approximate convex pieces
3: $\quad$ **for** each $d_{ik} \in D_i$ **do**
4: $\quad\quad ch_{ik} = CH(d_{ik})$
5: $\quad$ **end for**
6: **end for**
7: $P \leftarrow \emptyset$
8: **for** each i **do**
9: $\quad$ **for** each $ch_{ik} \in CH_i$ **do**
10: $\quad\quad$ Uniformly sample points on ch_{ik}
11: $\quad\quad P \leftarrow$ points that are external to all ch_{jk} from all $m_j \in M$
12: $\quad$ **end for**
13: **end for**
14: return P

While many choices can be made for constructing the τ-surfaces and sampling them, we use Fast Approximate Convex Decomposition [15, 14] for the first phase (steps 1–6) and Uniform Obstacle-Based PRM (UOBPRM) [33] for the second phase (steps 8–13). Hence, generated samples are uniformly distributed with respect to τ-surfaces but "nearly" uniform with respect to the surface of the original model.

In the following, we describe how to use FACD to decompose the model into approximate convex pieces and UOBPRM to generate uniformly distributed points around the surfaces.

3.1 Decomposing Approximate Convex Pieces

Fast Approximate Convex Decomposition (FACD) [15, 14] is used to decompose the model into approximate convex pieces efficiently. It also generates the convex hulls of the decomposed parts along with the decomposed pieces.

Approximate Convex Decomposition (ACD) partitions a given object (2D or 3D) into nearly convex pieces such that the maximum concavity of each piece does not exceed a user defined threshold. The ACD algorithm in [22, 23, 21] finds the maximum concave vertex in the model and constructs a single cut containing this vertex. The algorithm is then applied recursively with the resultant components. Figure 1(a) shows the resulting decomposition for the dinopet model.

Fast Approximate Convex Decomposition (FACD) [15, 14] improves the algorithm in [22, 23, 21] in terms of quality and efficiency of the decomposition. The improvement in efficiency is observed due to reduced recursion depth. Instead of using a single cut to decompose the model or a component at every step in the recursion, FACD uses multiple non-crossing cuts simultaneously to partition the model or a component into more than two components. Also, FACD introduces a new measure of concavity called relative concavity which ranks a cut based on its impact on the concavity of the model. More specifically, it is the ratio of the concavity of the model before and after the application of the cut. The user-defined tolerance is checked against the relative concavity measure to achieve a decomposition which prioritizes significant concavity change over gradual

(a) ACD [22] (b) FACD [15]

Figure 1: Decomposition of the dinopet model. With ACD, the tolerance is set to 0.07 to highlight the fingers in the hands and feet causing unnecessary segmentation in the neck and toe regions. With FACD, segmentation of the fingers do not cause unnecessary segmentation in the neck and toe regions. Only FACD decomposes the tail.

increase or decrease on the application of a cut. As shown in the Figure 1(b), FACD decomposition does not decompose the neck of the dinopet model or the toe region while partitioning the fingers in the hands and the legs of the model. However, Figure 1(a) decomposes the neck to obtain a decomposition along the fingers and also fails to decompose the tail.

The FACD algorithm, given in Algorithm 2, proceeds in a bottom-up approach where a set of potential cuts are predicted and a subset of final cuts is picked from this potential cut set using the relative concavity measurement. The components formed on application of these final cuts are then processed recursively with the same algorithm until their concavity measurement is below the tolerance. For more details, please refer to [15, 14].

Algorithm 2 FACD(M,τ)

Input: A model M and tolerance τ.
Output: Components $\{M_i\}$ of decomposed M

```
Find the potential cuts {C_k} in M
Use τ to select a set of cuts {C_r} from {C_k}
Apply {C_r} to decompose M into components {M_i}
for each M_i in {M_i} do
  if concavity(M_i) ≥ τ then
    FACD(M_i,τ)
  end if
end for
```

3.2 Generating Uniformly Distributed Samples on Surfaces

Uniform Obstacle-Based PRM (UOBPRM) [33] is a PRM method which samples points near the obstacle surface by randomly placing fixed length line segments and testing for intersections with obstacle surfaces. The resulting samples are guaranteed to be uniformly distributed around the obstacle surface and are more efficient to compute than other obstacle-minded methods such as Gaussian [6] and Bridge-test [17] sampling. Algorithm 3 gives a brief sketch of UOBPRM sampling.

Algorithm 3 UOBPRM(M,n,l,t)

Input: A model M, a number of samples n, a line segment length l, and a step-size t.
Output: A set of points P uniformly distributed near M.

Set a bounding box whose margin is l away from M.
$P = \emptyset$
for $i = 1 \rightarrow n$ **do**
    Uniformly sample point c
    Generate a random direction $\vec{d}$
    Extend a segment s from c distance l in direction $\vec{d}$
    $P \leftarrow$ Intersect(s,l,t)
end for
return P

Algorithm 4 Intersect(s, l, t)

Input: A line segment s of length l and a step size t
Output: A set of intersections I

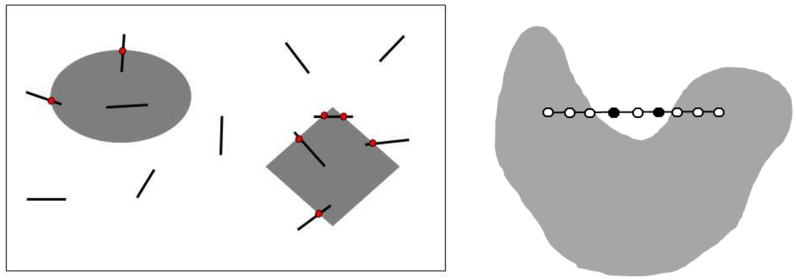
for $i = 1 \rightarrow (l/t)$ **do**
    Generate sample c_i along s
    if validity(c_i) $\neq$ validity(c_{i+1}) **then**
        Add the valid one to I
    end if
end for
return I

UOBPRM first uniformly samples a point c in C-space. Then, it creates a random line segment of user defined length l with c as one of its end-points along a random direction $\vec{d}$ from c. Figure 2(a) shows the resulting intersections (red) between the line segments (black) and the obstacles (gray).

To find the intersection of a line segment and the obstacle surface, a t length hop is taken over the the entire segment, checking validity at each hop. Algorithm 4 sketches the approach. Whenever a line segment intersects with an obstacle, there must be at least a valid and an invalid point across the crossing. Thus, UOBPRM checks for the change in validity at the hops and stores the valid point adjacent to an invalid point. The validity check is not terminated on finding a single sample as the line segment can have multiple intersections. For example in the Figure 2(b), there are two intersections of the line segment with the obstacle shown in gray, and hence two intermediate samples are retained. For more details about the algorithm, please see [33].

4 Results

We evaluate our approach in terms of efficiency and error in distance of the samples to the model surface. We manually chose the FACD tolerance which gives a visually meaningful decomposition along the model's major visible concavities. We studied the five different models shown in Figure 3. We compare UOBPRM on the original model, the convex hulls of the FACD decomposed parts, and the convex hull of the original model. The implementation is done in C++, and the experiments are conducted on a PC with Pentium CPU and 2GB RAM.



(a) UOBPRM Sampling (b) Finding Intersections

Figure 2: (a) UOBPRM sampling [33] where fixed length line segments are generated uniformly at random (black). The intersections (red) between these line segments and the obstacle surface (gray) are retained as valid samples. (b) Intersections of the line segment with the obstacle boundary are found by checking the validity of the intermediate points and retaining the valid one when there is a change in validity between two adjacent points [33].

4.1 Time Efficiency

Figure 4 compares the normalized time taken by UOBPRM to generate 500 samples near the surface of each of the models. Times are normalized to the time spent by UOBPRM on the original model. UOBPRM exhibits an exponential improvement in time for all the decomposed experimental models as compared to the original model. The maximum time improvement for every input model is achieved by using the convex hull of the original model. However, the improvement in time is dependent on the complexity of the model and its decomposition level. For example, the screwdriver model shows the most improvement while the pawn shows the least.

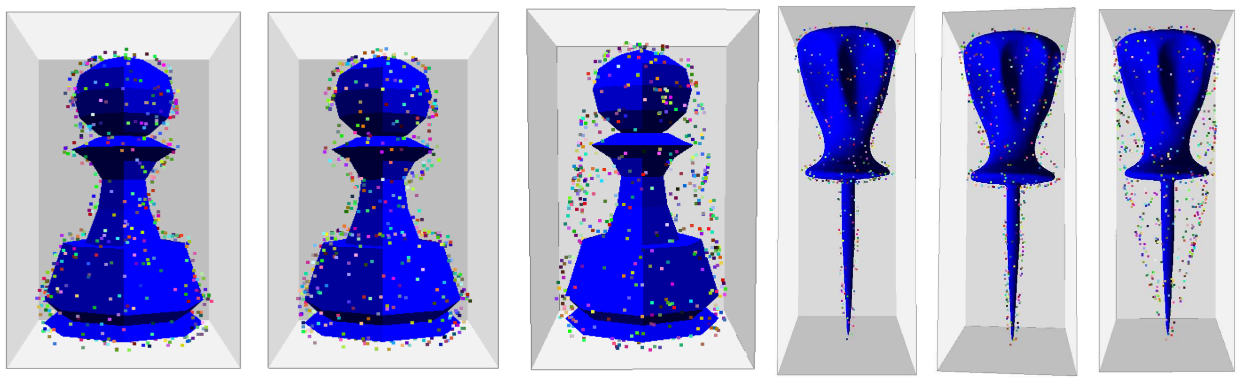
We next look at the efficiency of each method as a function of the number of requested samples. Figure 5(a) shows the results for the bird model (in Figures 3(m)–3(o)). As expected, the performance improvement is independent of the number of samples requested. Hence, given a particular decomposition, we will achieve same time improvement for any number of samples.

Figure 5(b) compares the normalized time using different FACD levels on the bird model. They are sorted by decreasing levels of decomposition, e.g., FACD-5 has 13 components and FACD-0 has 3 components. The results show that even for a high level of decomposition (e.g., FACD-5), we achieve faster sample generation as compared to the original model. However, as efficiency decreases with increase in decomposition levels, it is not advisable to use the exact convex decomposition.

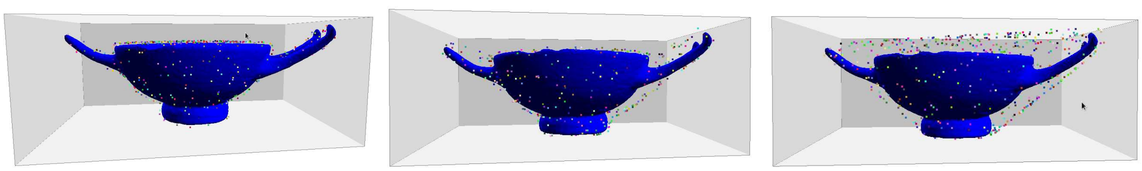
4.2 Distance Error due to Approximation

Using an approximate model, whether the convex hull or the convex hulls of the FACD components, introduces errors in the distribution, particularly in regions of high concavity. Figure 3 shows the resulting sample distribution for the five experimental models. The distribution using the FACD components is almost the same as using the original model, while simply using the convex hull of the original model has much larger differences. For example, consider the region between the bird's wings in Figure 3(m)–3(o). UOBPRM on FACD produces samples near the wing surface and head while UOBPRM on the convex hull has samples scattered through the entire trapezoidal region.

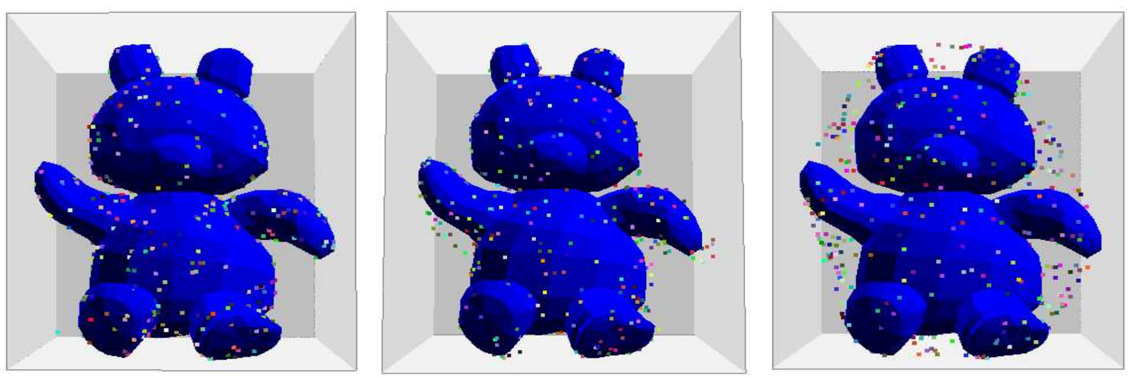
We quantify this error as the average distance of the samples to the surface of the original model. Figure 6(a) compares the normalized average distance of the samples generated using UOBPRM on the convex hulls of the decomposed model parts, on the convex hull of the original model, and on the original model for the five different input models. The error in distance from the surface



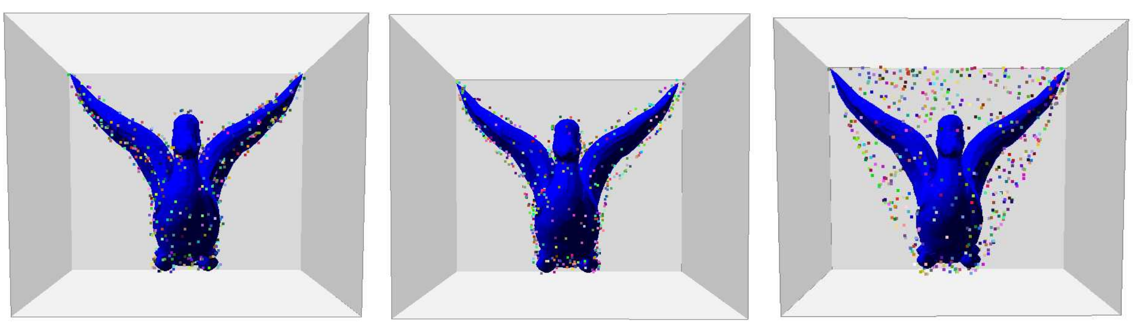
(a) Original (b) FACD (c) Convex Hull (d) Original (e) FACD (f) Con. Hull



(g) Original (h) FACD (i) Convex Hull



(j) Original (k) FACD (l) Convex Hull



(m) Original (n) FACD (o) Convex Hull

Figure 3: Point distribution over various models using UOBPRM on the original model, the convex hulls of FACD decomposed parts, and the convex hull of the original model. (a)-(c) are pawn models, (d)-(f) are screwdriver models, (g)-(i) are drum models, (j)-(l) are teddy models, and (m)-(o) are bird models.

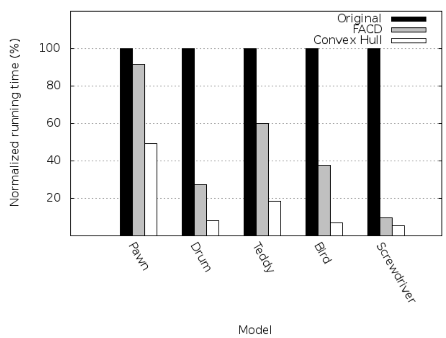
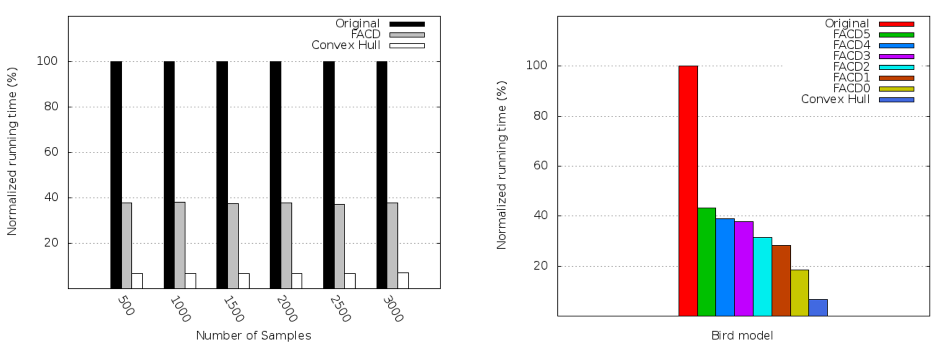


Figure 4: Time efficiency of UOBPRM with 500 samples over different models. The time is normalized to the time taken for original model.



(a) Varying the Number of Samples (b) Varying the FACD Level

Figure 5: Time efficiency of UOBPRM under different conditions on the bird model. All times are normalized to the time taken by the original model.

of the model is less if we use the convex hulls of the decomposed model parts than the error in distance due to convex hull of the original model. We also see in Figure 6(b) how the error changes when varying the FACD level for the bird model. (The levels are the same as in Figure 5(b).)

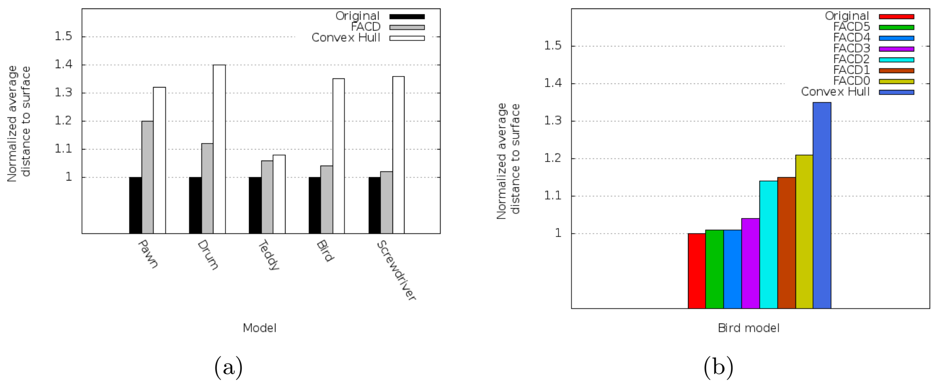


Figure 6: Average distance of 500 UOBPRM samples to the original model surface for (a) different input models and (b) the bird model with varying FACD levels. Average distance is normalized to UOBPRM on the original model.

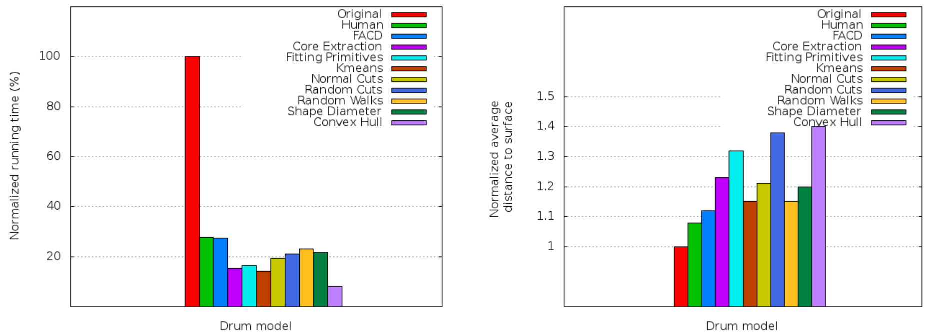
Using Figures 6(b) and 5(b), we see that the time improvement gained by decreasing the approximation level is at the expense of the distance error. Hence, an optimal solution would be to choose a trade-off between efficiency in time and error in distance to define the desired sample distribution around a complex model.

4.3 Comparison to Other Decomposition Algorithms

We next compare FACD to the seven decomposition algorithms given by the Princeton Mesh Segmentation Benchmark [8] as well as to a human decomposition. We applied UOBPRM on the convex hulls of the decomposed parts given by the various segmentations. To obtain a fair comparison, we chose the decomposition level in all the different methods and human segmentation to yield the same number of components as FACD. However, certain methods, like Shape Diameter, had a single level of decomposition. Figure 7 compares the time and error in distance for FACD, the seven segmentation methods, the human given segmentation, and the original model's convex hull to the original model. The results show that FACD compares similarly to the segmentation provided by humans. Although FACD does not perform better with respect to time than the other computational methods, it has the least error in distance among all of them. Hence, FACD is better at approximating the model than any other computational segmentation algorithm.

4.4 Application to Motion Planning

Finally, we apply the approach to solve a motion planning problem where a robot must navigate between the legs of two octopuses, see Figure 8. The bounding box is constrained so the only way for the robot to find a path from the bottom of the environment to the top is to pass through the free area among the legs. Simply using the convex hull to approximate the octopus's shape would exclude any sampling around the legs, the only passage available between the top and bottom of the environment. It would not be possible to solve the problem using such an approximation. However, the FACD model gives space between the legs for the robot to move. The solution path given in



(a) (b)

Figure 7: (a) Time efficiency and (b) average distance of 500 UOBPRM samples on the decomposed drum model using FACD, the human segmentation, and the other computational segmentations from Princeton Benchmark [8]. Times are normalized to the time taken by the original model. Average distances are normalized to that of the original model.

Figure 8 as the pink swept volume takes 13.48 sec to solve. Conversely, the environment with covex hull models cannot solve the problem even after 1 hour of computation.

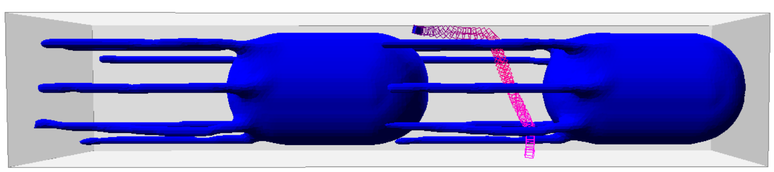


Figure 8: The robot must pass between the legs of the octopus to find a path from the bottom of the environment to the top of the environment. The problem is impossible to solve using convex hull approximations but can be easily solved with the FACD model.

5 Summary

We present an algorithm for "nearly" uniform sampling on surfaces that first decomposes the model into approximately convex surfaces and then generates uniform samples on the convex hulls of the decomposed parts or τ-surfaces. We use Fast Approximate Convex Decomposition (FACD) to partition the surfaces into pieces whose concavity is guaranteed to not exceed a threshold τ. We then use Uniform Obstacle-Based Sampling (UOBPRM) to generate uniform samples around the τ-surfaces. As validated by our results, our approach provides an fair trade-off between uniformity of distribution and efficiency. Also it has the least error in average distance when compared to other decomposition algorithms from the Princeton Benchmark [8]. Our method is able to solve a motion planning problem with complex geometric models as obstacles where simply using convex hulls would fail.

6 Acknowledgement

This research supported in part by NSF awards CRI-0551685, CCF-0833199, CCF-0830753, IIS-096053, IIS-0917266, NSF/DNDO award 2008-DN-077-ARI018-02, by the DOE NNSA under the

Predictive Science Academic Alliances Program by grant DE-FC52-08NA28616, by THECB NHARP award 000512-0097-2009, by Chevron, IBM, Intel, Oracle/Sun and by Award KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST). This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva, *Computing and rendering point set surfaces*, IEEE Transactions on Visualization and Computer Graphics **9** (2003), no. 1, 3–15.

[2] N. M. Amato and Y. Wu, *A randomized roadmap method for path and manipulation planning*, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), 1996, pp. 113–120.

[3] Nancy M. Amato, O. Burchan Bayazit, Lucia K. Dale, Christopher Jones, and Daniel Vallejo, *Obprm: an obstacle-based prm for 3d workspaces*, Proceedings of the third workshop on the algorithmic foundations of robotics on Robotics : the algorithmic perspective: the algorithmic perspective (Natick, MA, USA), WAFR '98, A. K. Peters, Ltd., 1998, pp. 155–168.

[4] Masaki Aono, Paolo Denti, David E. Breen, and Michael J. Wozny, *Fitting a woven cloth model to a curved surface: Dart insertion*, IEEE Comput. Graph. Appl. **16** (1996), no. 5, 60–70.

[5] J. Barraquand and J. C. Latombe, *Robot motion planning: A distributed representation approach*, Int. J. Robot. Res. **10** (1991), no. 6, 628–649.

[6] V. Boor, M. H. Overmars, and A. F. van der Stappen, *The Gaussian sampling strategy for probabilistic roadmap planners*, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), vol. 2, May 1999, pp. 1018–1023.

[7] Juan Cao, Yitian Zhao, Ran Song, and Yingchun Zhang, *Local and global point sampling for structured point cloud simplification*, Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on, May 2012, pp. 1843 –1847.

[8] Xiaobai Chen, Aleksey Golovinskiy, and Thomas Funkhouser, *A benchmark for 3D mesh segmentation*, ACM Transactions on Graphics (Proc. SIGGRAPH) **28** (2009), no. 3.

[9] Yong Chen and Charlie C. L. Wang, *Uniform offsetting of polygonal model based on layered depth-normal images*, Comput. Aided Des. **43** (2011), no. 1, 31–46.

[10] Yong Chen, Hongqing Wang, David W. Rosen, and Jarek Rossignac, *A point-based offsetting method of polygonal meshes*.

[11] David Cline, Stefan Jeschke, Anshuman Razdan, Kenric White, and Peter Wonka, *Dart throwing on surfaces*, Computer Graphics Forum **28** (2009), no. 4, 1217–1226.

[12] Robert L. Cook, *Stochastic sampling in computer graphics*, ACM Trans. Graph. **5** (1986), no. 1, 51–72.

[13] Mohamed S. Ebeida and Scott A. Mitchell, *Uniform random voronoi meshes*, Proceedings of the 20th International Meshing Roundtable (WilliamRoshan Quadros, ed.), Springer Berlin Heidelberg, 2012, pp. 273–290.

[14] Mukulika Ghosh, *Fast approximate convex decomposition using relative concavity*, Master's thesis, Texas A& M University, 2012.

[15] Mukulika Ghosh, Nancy M. Amato, Yanyan Lu, and Jyh-Ming Lien, *Fast approximate convex decomposition using relative concavity*, Computer-Aided Design **45** (2013), no. 2, 494–504, Solid and Physical Modeling 2012.

[16] J. P. Grossman and William J. Dally, *Point sample rendering*, In Rendering Techniques 98, Springer, 1998, pp. 181–192.

[17] D. Hsu, T. Jiang, J.H. Reif, and Z. Sun, *Bridge test for sampling narrow passages with probabilistic roadmap planners*, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), 2003, pp. 4420–4426.

[18] Philip M. Hubbard, *Approximating polyhedra with spheres for time-critical collision detection*, ACM Trans. Graph. **15** (1996), no. 3, 179–210.

[19] L. Kavraki and J. C. Latombe, *Randomized preprocessing of configuration space for fast path planning*, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), 1994, pp. 2138–2145.

[20] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars, *Probabilistic roadmaps for path planning in high-dimensional configuration spaces*, IEEE Trans. Robot. Automat. **12** (1996), no. 4, 566–580.

[21] Jyh-Ming Lien, *Approximate convex decomposition and its application*, Ph.D. thesis, Texas A& M University, 2006.

[22] Jyh-Ming Lien and Nancy M. Amato, *Approximate convex decomposition of polyhedra*, Proceedings of the 2007 ACM Symposium on Solid and Physical Modeling (SPM'07) (Beijing, China), 2007, pp. 121–131.

[23] Jyh-Ming Lien and Nancy M. Amato, *Approximate convex decomposition of polyhedra and its applications*, Computer Aided Geometric Design **25** (2008), no. 7, 503 – 522, Selected papers from the Solid and Physical Modeling and Applications Symposium 2007 (SPM 2007).

[24] Jyh-Ming Lien and Yanyan Lu, *Planning motion in point-represented contact spaces using approximate star-shaped decomposition*, Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems (Piscataway, NJ, USA), IROS'09, IEEE Press, 2009, pp. 5420–5425.

[25] Rong Liu, Hao Zhang, and James Busby, *Convex hull covering of polygonal scenes for accurate collision detection in games*, Proceedings of graphics interface 2008 (Toronto, Ont., Canada, Canada), GI '08, Canadian Information Processing Society, 2008, pp. 203–210.

[26] S. Lloyd, *Least squares quantization in pcm*, IEEE Trans. Inf. Theor. **28** (2006), no. 2, 129–137.

[27] Diego Nehab and Philip Shilane, *Stratified point sampling of 3d models*, Proceedings of the First Eurographics conference on Point-Based Graphics (Aire-la-Ville, Switzerland, Switzerland), SPBG'04, Eurographics Association, 2004, pp. 49–56.

[28] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin, *Matching 3d models with shape distributions*, Proceedings of the International Conference on Shape Modeling & Applications (Washington, DC, USA), SMI '01, IEEE Computer Society, 2001, pp. 154–.

[29] M. H. Overmars and P. Švestka, *A probabilistic learning approach to motion planning*, Algorithmic Foundations of Robotics, A. K. Peters, Wellesley, MA, 1995.

[30] Greg Turk, *Re-tiling polygonal surfaces*, SIGGRAPH Comput. Graph. **26** (1992), no. 2, 55–64.

[31] J.P. van den Berg and M.H. Overmars, *Using workspace information as a guide to non-uniform sampling in probabilistic roadmap planners*, Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, vol. 1, May 2004, pp. 453 – 460 Vol.1.

[32] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, *MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space*, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), vol. 2, 1999, pp. 1024–1031.

[33] Hsin-Yi (Cindy) Yeh, Shawna Thomas, David Eppstein, and Nancy M. Amato, *Uobprm: A uniformly distributed obstacle-based prm*, Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS) (Vilamoura, Algarve, Portugal), 2012.