ME 145

Robotic Planning and Kinematics

Lab Session No. 5

Instructions

Submit your code through iLearn. Your code and reports are due on Monday, May 20th, 11:59PM. No late submission will be accepted.

E4.3 Programming: Sampling algorithms (40 points).

Consider the unit square [0, 1]2 in the plane. Pick and arbitrary integer k and do:

- (5 points) write formulas for the n = k2 sample points in the uniform Sukharev center grid;
- (5 points) write formulas for the n = k2 sample points in the uniform corner grid;
- (30 points) write the following programs (representing a grid with n entries in [0,1]2 by a matrix with n rows and 2 columns):

computeGridSukharev (10 points).

Input: the number of samples n (assuming $n = k^2$ for some integer number k).

Output: the uniform Sukharev center grid on $[0, 1]^2$ with $n^{1/2}$ samples along each axis.

computeGridRandom (10 points).

Input: the number of samples n.

Output: a random grid on $[0, 1]^2$ with n uniformly-generated samples.

computeGridHalton (20 points).

Input: the number of samples n; two prime numbers b1 and b2.

Output: a Halton sequence of n samples inside $[0, 1]^2$ generated by the two prime numbers b1 and b2

For each function, do the following:

- Explain how to implement the function, possibly deriving analytic formulas, and characterize special cases,

- Program the function, including correctness checks on the input data and appropriate error messages, and
- Verify your function is correct by plotting the three grids for n = 100.

<u>E4.4 Programming: Collision detection primitives</u> (45 points).

Write the following programs:

<u>isPointInConvexPolygon</u> (15 points).

Input: a point q and a convex polygon P .

Output: true (1) or false (0)

<u>doTwoSegmentsIntersect</u> (15 points).

Input: two segments described by their respective vertices p1, p2, and p3, p4.

Output: true (1) or false (0). If true, then return also the intersection point.

<u>doTwoConvexPolygonsIntersect</u> (15 points).

Input: two convex polygons P1 and P2.

Output: true (1) or false (0).

<u>For each function, do the following:</u>

- Explain how to implement the function, possibly deriving analytic formulas, and characterize special cases; specifically, write a pseudo-code routine to check whether a point is inside a convex polygon,
- Program the function, including correctness checks on the input data and appropriate error messages, and
- Verify your function is correct on a broad range of test inputs.