This exam is closed book and closed note. You may use any algorithms or results given in class or in the Mount lecture notes. We do not expect proofs, but do expect you to support answers when asked.

The boxes here are for Gradescope. Put your primary answer in each box. If you have supporting comments, scratch work, or other, put it on other blank sections and we will be able to see and take it into account. If a blank section is small is means the answer is short, but not the reverse.

I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Topic</th>
<th>Worth</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Short answer</td>
<td>/20</td>
</tr>
<tr>
<td>2</td>
<td>Vector operations</td>
<td>/15</td>
</tr>
<tr>
<td>3</td>
<td>Colliders</td>
<td>/20</td>
</tr>
<tr>
<td>4</td>
<td>Vehicle orientation</td>
<td>/15</td>
</tr>
<tr>
<td>5</td>
<td>AABBs and KDOPs</td>
<td>/15</td>
</tr>
<tr>
<td>6</td>
<td>Navmesh</td>
<td>/15</td>
</tr>
</tbody>
</table>
**Problem 1. Short answer** (20 pts, 3-4 each). Explanations are not needed, but may be given for partial credit, or to insure we understand your answer.

a) *Event loop.* Event Loops are (put your answer in the box).

- a. A bad movie about time loops
- b. A sugary cereal
- c. Recurring appointments
- d. A way to organize interactive programs

b) *Event loops.* In Unity, what's the difference between the FixedUpdate and Update calls in the event loop?

- **Update** is called once per frame, which can vary
- **FixedUpdate** is called on a fixed time interval


c) *Device interaction.* What is the difference between input by device polling and device events?

- In polling, the program interrogates the device
- In events, the device interrupts the program @ events

d) *Perp vector.* Given a vector \( \vec{v} = \langle x, y \rangle \), define a perp vector that is 90 degrees clockwise from \( \vec{v} \).

\[
M \times \vec{v} = \begin{bmatrix} \cos 90 & -\sin 90 \\ \sin 90 & \cos 90 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -y \\ x \end{bmatrix}
\]

e) *Capsule collider.* What data do you need to define a capsule collider?

- Two points and a *radius*
Problem 2. Vector operations (15 pts).
Let’s assume zombies never turn their heads, and they only attack those they see in front of them. So if you’re behind them you’re safe; if you’re in front, watch out.

We’re represent a zombie as standing with arms out, and the three points $p_1$, $p_2$, and $p_3$, on the zombies head, left hand and right hand respectively as in the diagram. Points $p_2$ and $p_3$ are symmetric about the neck, and at the same height.

Use $p_1$, $p_2$, and $p_3$, define a 3D zombie coordinate system centered between $p_2$ and $p_3$, with an origin point $p_0$, a forward vector $\vec{n}$ that goes towards the front, an up vector $\vec{u}$ that goes up through the head, and a right vector $\vec{v}$ towards the right hand, with $\vec{v} = (p_2 - p_3)/|p_2 - p_3|

a) Give the origin point $p_0$.
\[ p_0 = \frac{p_2 + p_3}{2} \]

b) Give the forward vector $\vec{n}$ using only $p_1$, $p_2$, and $p_3$.
\[ \vec{n} = (p_3 - p_2) \times (p_1 - p_2) \]

b) Give equations for the vector $\vec{u}$ in two ways.
\[ \vec{u} = p_1 - p_0 = \vec{n} \times (p_3 - p_2) \]

c) Assume you are located at point $q$. Give a geometric test to determine whether the zombie is facing you. In particular, determine if $q$ lies in front or in back of the plane through the zombie.

Let $c = \vec{n} \cdot (q - p_0)$
If $c > 0$, faces you
If $c < 0$, faces away
If $c = 0$, faces @ 90 degrees
Problem 3. Line-sphere collider (15 points). Feeling like a grinch, you create a game to shoot down birthday party balloons with a laser cannon. The laser is instantaneous – if the ray it creates hits the balloon, pop. The laser is given by a point \( p \) and a vector \( \vec{v} \), the balloon by a center \( C \) and a radius \( R \). Figure out a collider for the laser cannon and balloon. And, do this in coordinate free notation so the answer applies in 2 or 3 dimensions. Assume that \( |\vec{v}| = 1 \) although in the diagram it is long.

(a) First give an equation for the parametric point-vector form of the laser cannon ray, complete with the valid range of the parameter \( t \).

\[
\rho(t) = p + t\vec{v}, \quad t \in [0, +\infty]
\]

(b) Decompose the vector from \( P \) to \( C \) into two orthogonal components along, and perpendicular, to the vector \( \vec{v} \).

\[
\vec{v} = C - p
\]
\[
\vec{v}_1 = (\vec{v} \cdot \vec{v})\vec{v} \quad \begin{array}{c}
\text{normalized because} \\
|\vec{v}_1| = 1
\end{array}
\]
\[
\vec{v}_2 = \vec{v} - \vec{v}_1
\]

(b) Finally use the results of (b) to create a test for the intersection of the ray and balloon.

\[
\text{if } |\vec{v}_2| < R, \text{ intersect} \\
\text{else not} \\
( |\vec{v}_2| = R \text{ case not critical} )
\]
Problem 4. KDOPs and AABBs (20 points). For collision detection, a discrete oriented polytope (DOP) generalizes the idea of axis aligned bounding boxes (AABB). The AABB is represented by the maximum and minimum along two independent axes, x and y (or x,y,z in 3D). A KDOP uses more axes.

If an AABB is represented by four values (xmin,xmax,ymin,ymax), how would you perform the union and intersection of two AABBs? Assume the two are A and B, and the values are A.xmin, A.xmax, etc, how do you compute the union and intersection?

\begin{align*}
\text{a) Union} \\
\text{Union.min} &= \min(A.xmin, B.xmin) \\
\text{Union.max} &= \max(A.xmax, B.xmax) \\
\text{b) Intersection} \\
\text{Intersect.min} &= \max(A.xmin, B.xmin) \\
\text{Intersect.max} &= \min(A.xmax, B.xmax) \\
\text{c) Given the results of the Intersection, how can you tell the intersection is the null set, ie, the two AABBs don't overlap? Give a boolean expression that is true if the intersection is empty.}
\end{align*}
Generalizing to 4DOPs, if one is represented by eight values, assuming the new axes are w and z, we have (xmin,xmax,ymin,ymax,wmin,wmax,zmin,zmax) with (wmin,wmax) the min and max distance in the NE-SW direction (+1 slope), and (zmin,zmax) the min and max distance in the NW-SE direction (-1 slope).

\[ \text{A}\bigcup\text{B}\quad \text{A INTERSECTION B} \]

d) How would you generalize your solution to the AABB intersection to work for 4DOPs?

You add similar operations to the intersection along the w and z axes, and then add boolean expressions for the w \& z axes.

e) How would you compute the max and min distances along an arbitrary axis defined by the vector \( \vec{v} \) given a set of points? Give a short algorithm.

\[ \vec{v} \cdot (p - o) = d \text{ is the distance of } p \text{ along the } \vec{v} \text{ axis, with } o \text{ the origin (center of the points works)} \]

Find the max and min of these points distance \( d \) along \( \vec{v} \).
Points \( p' \) that point away from \( \vec{v} \) will yield the minimum.
Problem 5. Forward kinematics for animation (15 points). Since this is Maryland we’re going to animate a crab. Just one leg, since the exam is too short for eight. The resting pose is one that puts the leg on the ground, with the coordinate systems for each joint rotated 45 degrees from the ground plane.

We have that:
- a) \( j_1 \) is 7 units up along \( y \) from \( j_0 \) and rotated 90 degrees clockwise
- b) \( j_2 \) is 8 units along \( x \) from \( j_1 \)
- c) The tip of the foot at \( P \) is 4 units along \( x \) from \( j_2 \)

(a) Express the position of the point \( P \) in the bind pose with respect to each joint

\[
P_{[j2]} = \begin{bmatrix} 4 \\ 0 \\ 1 \end{bmatrix} \quad P_{[j1]} = \begin{bmatrix} 12 \\ 0 \\ 1 \end{bmatrix} \quad P_{[j0]} = \begin{bmatrix} 7 \\ -12 \\ 1 \end{bmatrix}
\]

(b) Express the following local pose transformations, each as a homogeneous 3 x 3 matrix assuming the bind pose as given.

\[
T_{[j1\rightarrow j2]} = \begin{bmatrix} 1 & 0 & 8 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad T_{[j0\rightarrow j1]} = \begin{bmatrix} 0 & 1 & 7 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}
\]
Problem 5. Forward kinematics for animation (15 points). Since this is Maryland we're going to animate a crab. Just one leg, since the exam is too short for eight. The resting pose is one that puts the leg on the ground, with the coordinate systems for each joint rotated 45 degrees from the ground plane.

We have that:

a) $j_1$ is 7 units up along y from $j_0$ and rotated 90 degrees clockwise
b) $j_2$ is 8 units along x from $j_1$
c) The tip of the foot at P is 4 units along x from $j_2$

(a) Express the position of the point P in the bind pose with respect to each joint

\[
P_{[j_2]} = \begin{bmatrix} 4 \\ 0 \\ 1 \end{bmatrix} \quad P_{[j_1]} = \begin{bmatrix} 12 \\ 0 \\ 1 \end{bmatrix} \quad P_{[j_0]} = \begin{bmatrix} 0 \\ 7 \\ 1 \end{bmatrix}
\]

(b) Express the following local pose transformations, each as a homogeneous 3 x 3 matrix assuming the bind pose as given.

\[
T_{[j_1-j_2]} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad T_{[j_0-j_1]} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 7 \\ 0 & 0 & 1 \end{bmatrix}
\]
(c) To set up the last problem we define $T(dx,dy)$ as a homogeneous translation matrix, and $Rot(\theta)$ as homogeneous rotation matrix. Assume that we have the following rotations at each joint:

- The joint $j0$ lifts the leg by rotating 20 degrees counterclockwise.
- Joint $j1$ revolves counterclockwise by an angle of 10 degrees.
- Joint $j0$ revolves clockwise by an angle of 5 degrees.

Express a combined homogeneous transformation matrix that maps $p_{j2}$ to $p_{j0}$ under the individual transformations above. Give it as a sequence of transformations. Just represent them as $T(dx,dy)$ and $Rot(\theta)$ – don't give full matrices.

$$
M = Rot (20) \ast Rot (5) \ast \overrightarrow{T_{j0\rightarrow j1}} \ast R(10) \\
\ast \overrightarrow{T_{j1\rightarrow j2}} \ast R
$$

If you assume last rotation (Third one above) is around $\mathbf{z}$-

$$
M = Rot (30) \ast \overrightarrow{T_{j0\rightarrow j1}} \ast R(10) \\
\ast \overrightarrow{T_{j1\rightarrow j2}} \ast R(5) \ast R
$$
Problem 6. Navmesh (15 points). In creating a Navmesh from a polygon we used the “bunny ear” algorithm to triangulate a simple polygon by adding chords. Here are two examples with a polygon of \( n = 10 \) vertices yielding \( t=8 \) resulting triangles, and \( c=7 \) resulting chords.

(a) Give a formula for the number of triangles \( t \) for the general case of \( n \) vertices. This will be a formula with \( t \) as a function of \( n \).

\[
 t = n - 2 \\
\]

By induction (informally):
Base: \( n = 1 \), one triangle
Induction: Add pt, add one tri.

(b) Give a formula for the number of chords \( c \) for the general case of \( n \) vertices. This will be a formula with \( c \) as a function of \( n \).

\[
 c = n - 3 \\
\]

By induction:
Base: \( n = 3 \), zero chords
Induction: Add pt, add one chord.

(c) For a vertex to be cut off as part of an ear, it must “stick out” of the original polygon. Vertices that “stick in” can’t be cut off. We can call the first convex, the latter concave. In the diagram above one vertex is marked as in, one as out. If the polygon vertices are ordered clockwise, how can you compute for each vertex whether it is an in or out vertex. (Hint: think vector operations).

Assume vertex \( V_n \) has vertices \( V_{n-1}, V_{n+1} \) on either side, mod \( n \) (so they wrap around.)

\[
 a \quad V_{n+1} \\
 V_n \\
 V_{n-1} \\
\]

Then \( v = (V_{n+1} - V_n) \times (V_n - V_{n-1}) \) will stick up out of the plane (ie, be positive) if the vertex \( V_n \) is out, and be negative if in.
Scratch paper
If you want us to check anything here please make a note at the original question.