2IV10/2IV60 Computer Graphics

Intermediate Examination, December 16 2013, 10:45 – 12:30

This examination consist of **three** questions with in total 9 subquestion. Each subquestion weighs equally. In all cases: **EXPLAIN YOUR ANSWER. Use sketches where needed to clarify your answer. Read first all questions completely.** If an algorithm is asked, then a description in steps or pseudo-code is expected, which is clear enough to be easily transferred to real code. Aim at compactness and clarity. Use additional functions and procedures if desired. Give from each function and procedure a short description of input and output. The use of the book, copies of slides, notes and other material is not allowed.

1 We consider drawing a cylinder. Suppose that we have a basic function *DrawCyl* , which draws a cylinder defined by $x^2 + y^2 = 1$ and $0 \leq z \leq 1$: unit radius, unit height, and the axis of rotational symmetry is aligned with the z-axis. This cylinder is defined in a local coordinate frame, global positions X are derived from local coordinates X' by $X=MX'$, where M is a homogenous transformation matrix. Furthermore, suppose that we have a set of functions to define transformation matrices, *i.e.*,

- $T(V)$ gives a translation matrix over a vector V;
- $S(s_x, s_y, s_z)$ gives a scaling matrix with scale factors s_x, s_y, and s_z for the axes; and
- $R_k(a)$ gives a counterclockwise rotation over a radians around axis k (= x, y, or z).

We want to define a more generic function to draw a cylinder. The center of the bottom of the cylinder should be a point P; the center of the top of the cylinder should be a point Q; the cylinder should have radius r. We define this function in a few steps. For b) and c) results from earlier subquestions can be used.

a) First, we make sure that the cylinder has the right dimensions. Define a matrix M_S such that setting $M = M_S$ gives the desired result.
Let $V = Q - P$. The height of the cylinder should become $|V|$ (= $\text{sqrt}(V_x^2 + V_y^2 + V_z^2)$), the radius must become r. Using a scaling matrix does the job: $M_s = S(r, r, |V|)$.

b) Next, define a matrix M_D such that the cylinder is drawn with the right orientation and dimensions. What is the angle α between the axis of the cylinder and the z-axis; what is the angle β between the projection of the axis of the cylinder on the XOY-plane and the X-axis? You can use these to define M_D such that setting $M = M_D$ gives the desired result.
Using spherical coordinates is convenient here to orient the cylinder, and the suggestions given hint at that. The angle α follows from $\alpha = \text{atan}(\text{sqrt}(V_x^2 + V_y^2) / V_z)$, the angle β is given by $\beta = \text{atan}(V_y / V_x)$. We can get the right orientation by first rotating around the y-axis over α, followed by a rotation around the global z-axis over β. Hence, $M_D = R_z(\beta)\, R_y(\alpha)\, M_S$.

c) Finally, define M such that a call to *DrawCyl* produces a cylinder from P to Q with radius r.
The only thing left over to do is to translate the cylinder such that the origin is moved to point P. Hence, $M = T(P)M_D$.

---

2 We consider making a movie of a human cannonball. The path $P(t)$ of the helmet of our volunteer is given by $P(t) = (30t,\ 0,\ 20t - 5t^2)$, where t is the time in seconds, and where $P_z(t) \geq 0$. We use two virtual cameras, one mounted on the helmet of the volunteer, and a fixed camera viewing the scene from the side. Both cameras are specified using an eye-point $E(t)$, a center-point $C(t)$, and a view-up vector $T(t)$. The opening angle is 90 degrees, both horizontally as well as vertically. Answer the following questions.

a) Give the distance D that is travelled and the maximum height H.

We first determine when the ground is hit. Here $P_z(t) = 0$, or $20t - 5t^2 = 0$, or $(20 - 5t)t = 0$. At $t = 0$ the flight starts, and at $t = 4$ the flight ends. Now, $D = P_x(4) - P_x(0) = 120$.

The maximum height is achieved when $P'_z(t) = 0$ (where the dash denotes differentiation with respect to time). Written out, this gives $20 - 10t = 0$, hence for $t = 2$ the maximum $H = P_z(2) = 20$ is achieved.

b) Give $E(t)$, $C(t)$, and $T(t)$ for the helmet mounted camera. Assume that the camera moves and views along the path of the helmet.

Obviously, $E(t) = P(t)$, as the camera moves along the path. The viewing direction must be tangent to the path. The tangent vector $P'(t)$ follows from straightforward differentiation: $P'(t) = (30, 0, 20 - 10t)$. The center-point can be set by adding $P'(t)$ to $E(t)$. If we want to fix the distance from view-point to center-point (not required according to the specification), we can use $C(t) = P(t) + d \; P'(t) \; / \mid P'(t) \mid$, where $d > 0$ is the distance.

For this case, $T(t)$ can be set to the standard value of $(0, 0, 1)$, assuming that the cannonball does not roll along its axis. It was not requested, but we can make this robust for wilder paths $P(t)$, including loopings, by using $T(t) = (-P'_z(t), 0, P'_x(t))$. This view-up vector is a 90 degrees rotated version of $P'(t)$.

c) Give E, C, and T for the fixed camera. Assume that a standard right-handed coordinate system is used. The camera is located on a tower, such that the flight is maximally visible. Make sure that the volunteer flies from left to right on the view by the camera.

If we look from the side, we see that the path is contained in a rectangle in the y=0 plane, with $0 \le x \le D$ and $0 \le z \le H$. For the centerpoint, we therefore use $C = (D/2, 0, H/2)$, and also, we pick $E_x = D/2$ and $E_z = H/2$. Next, we consider where to put the camera with respect to the y-direction. We have calculated that $D > H$, hence we must take D into account to get the complete scene in view. A sketch of the scene from top reveals that for the given opening angle of the camera a distance $D/2$ from the y=0 plane should be used. Also, we must view from the negative side, to make sure that we see the motion from left to right. Combining this gives $E = (D/2, -D/2, H/2) = (60, -60, 10)$. For the view-up vector we use straightforward $T = (0, 0, 1)$.

Not requested, but we furthermore can consider what distance from the y=0 plane to use for an arbitrary opening angle α. The relation to be satisfied is $\tan(\alpha/2) = (D/2) / d$, hence

$d = D / (2 \tan(\alpha/2))$. If α is 90 degrees, we get $d = D / 2$, for lower values of α we get higher values of d.

---

3 We consider a surface patch defined by $z = yx^2$, with $0 \le x \le 1$ and $0 \le y \le 1$.

a) Give a parametric definition $P(u, v)$ with $0 \le u, v \le 1$ of this surface; and an implicit definition $f(x, y, z) = 0$ of the unbounded surface where this patch is part of.
We can use x and y as u and v parameters. This gives $P(u, v) = (u, v, vu^2)$. Here the domains for x and y on one hand and u and v on the other hand matched nicely. If for instance the range $-1 \le x \le 1$ was specified, one would have to use $x = 2u - 1$. What would be needed for an arbitrary range $a \le x \le b$?
The implicit definition can simply be obtained by moving all terms to one side of the equation: $f(x, y, z) = z - yx^2$. An alternative is to use $f(x, y, z) = yx^2 - z$, but the first one is somewhat preferable, assuming that for high values of z we are outside the shape and that we use the convention to use positive values for f here.

b) Give a procedure to draw the surface patch, assuming a function $DT(A, B, C)$ is available to draw a triangle ABC. Use a parameter M for the number of segments that is used to approximate each side of the boundaries.
One of the standard recipes can be followed, for instance:

```
du := 1.0 / M; // Calculate increments in u and v direction.
dv := 1.0 / M;

for j := 0 to M − 1 do // Loop over j, producing strips with constant v
for i := 0 to M − 1 do // Loop over i, to produce the strips with triangles
{
    // Process a square region [u, u+du] × [v, v+dv] in the domain of the parameter space.

    u := i * du; // Calculate u and v
    v := j * dv;

   DT(P(u, v), P(u + du, v), P(u + du, v + dv)); // Draw a lower triangle
   DT(P(u, v), P(u + du, v + dv), P(u, v + dv)); // Draw an upper triangle
}
```

By far not the most efficient, but does the job.

c) Give a unit normal N for an arbitrary point on this surface patch. Choose yourself if you assume the point to be given via coordinates (x, y, z) or parameter values (u, v).

If we start from the parametric definition $P(u, v) = (u, v, vu^2)$, we can find a normal $N(u, v)$ via:

$$\begin{aligned} N(u, v) &= \partial P/\partial u \times \partial P/\partial v \\ &= (1, 0, 2uv) \times (0, 1, u^2) \\ &= (-2uv, -u^2, 1). \end{aligned}$$

Starting with the implicit definition $f(x, y, z) = z - yx^2$, we can find a normal $N(x, y, z)$ via:

$$\begin{aligned} N(x,y,z) &= \nabla f(x, y, z) \\ &= (-2xy, -x^2, 1). \end{aligned}$$

Because x and u, as well as y and v nicely match here, we can easily see that the two approaches give the same result.

To obtain unit normals, both have to be normalized to unit length:

$$\begin{aligned} N'(u, v) &= N(u, v) / |N(u, v)| \\ &= (-2uv, -u^2, 1) / \text{sqrt}(4u^2v^2 + u^4 + 1) \end{aligned}$$

and similarly:

$$\begin{aligned} N'(x,y,z) &= N(x, y,z) / |N(x, y, z)| \\ &= (-2xy, -x^2, 1) / \text{sqrt}(4x^2y^2 + x^4 + 1) \end{aligned}$$