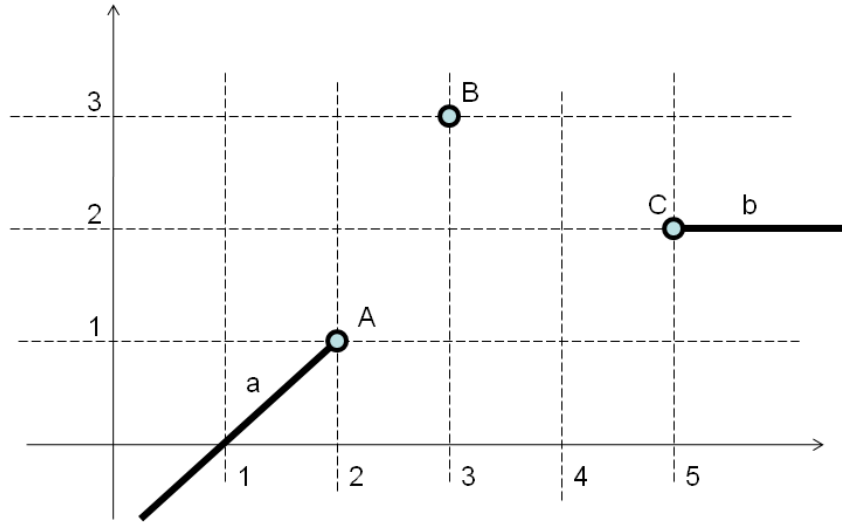


Transformaciones Geométricas, Curvas y Superficies

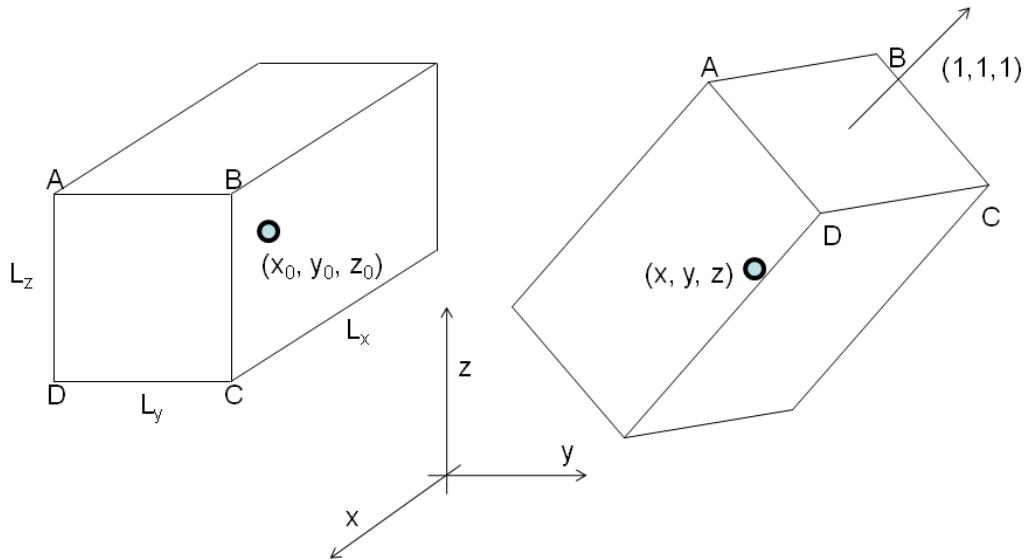
Ejercicio 1. (5 puntos)

Diseñar una curva con continuidad C^1 que una los segmentos a y b, pasando por los puntos A, B y C. Hay dos opciones principalmente para hacerlo. ¿Cuáles son? Elegir la opción que sea computacionalmente más simple. ¿Qué desventaja tiene?



Ejercicio 2. (5 puntos)

El prisma de la figura tiene dimensiones (L_x, L_y, L_z) y su centro está en la posición (x_0, y_0, z_0) . Queremos hacerlo el doble de grande, posicionarlo en (x, y, z) , y con la cara ABCD orientada en la dirección $(1, 1, 1)$ como se indica en la figura. Los segmentos AB y CD siguen siendo paralelos al plano XY. Calcular la transformación a realizar.



Nota: Utilizar dibujos que ayuden a aclarar los distintos pasos efectuados. No es necesario efectuar las multiplicaciones entre matrices al expresar la transformación final, es suficiente con expresar cada matriz y dejar el producto indicado.

OpenGL

Ejercicio 1. (3 puntos)

Queremos pintar en OpenGL un sistema solar, cuyos objetos (estrellas, planetas, satélites, etc.) están definidos de manera recursiva, con una transformación respecto a su padre. El objetivo del ejercicio es escribir el código de la función `Planet::DrawRecursive()`, para que todo el pintado sea eficiente y modular, poniendo especial énfasis en el manejo de la pila de OpenGL.

La clase `Planet` está definida de la siguiente manera:

```
class Planet {
    GLfloat transform[16]; //Transformación respecto al padre
    vector<Planet*> children; //Vector de punteros a los hijos
    void Draw(void); //Función que pinta el planeta (¡Conocida!)
    void DrawRecursive(void); //Función a programar
}
```

La función `display` hace sólo lo siguiente:

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
sun->DrawRecursive();
```

Para acceder a los hijos:

```
unsigned int size = children.size(); //Tamaño del vector
Planet* child = children[i]; //Hijo en la posición i del vector
```

Para definir una transformación:

```
glMultMatrix(transform); //Multiplica MODELVIEW por transform
```

Ejercicio 2. (4 puntos)

Apartado 2.a

¿Sobre qué primitiva(s) se aplica el modelo de iluminación de Phong en OpenGL?

¿Sobre qué primitiva(s) se aplica el sombreado de Gouraud (suave) en OpenGL?

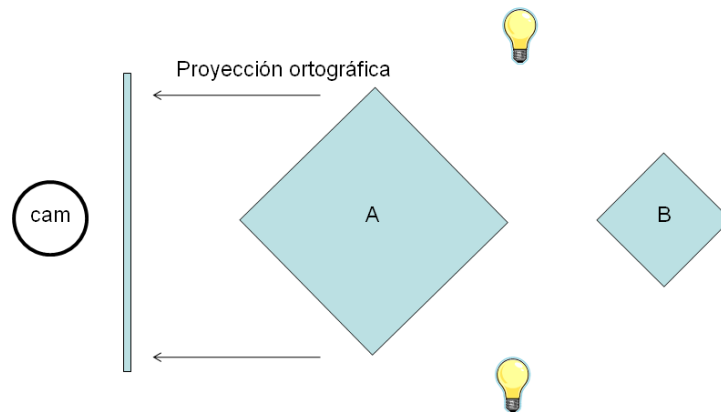
Apartado 2.b

El siguiente ejemplo 2D consta de 2 objetos opacos y 2 luces. Cada objeto tiene 4 segmentos. Se utiliza proyección ortográfica, y los segmentos del objeto A ocupan 10 pixels en pantalla, mientras que los segmentos del objeto B ocupan 5 pixels.

¿Cuántas veces se evalúan el modelo de iluminación de Phong y el modelo de sombreado de Gouraud si se pinta primero A y luego B?

¿Cuántas veces se evalúan el modelo de iluminación de Phong y el modelo de sombreado de Gouraud si se pinta primero B y luego A?

Indicar cómo se llega al resultado en cada caso.



Ejercicio 3. (3 puntos)

Estas dos imágenes han sido generadas con la misma luz, el mismo objeto, y los mismos parámetros de OpenGL. Lo único que cambia es la rotación del objeto. ¿Por qué la imagen de la derecha tiene un brillo y la de la izquierda no? Explícalo.

