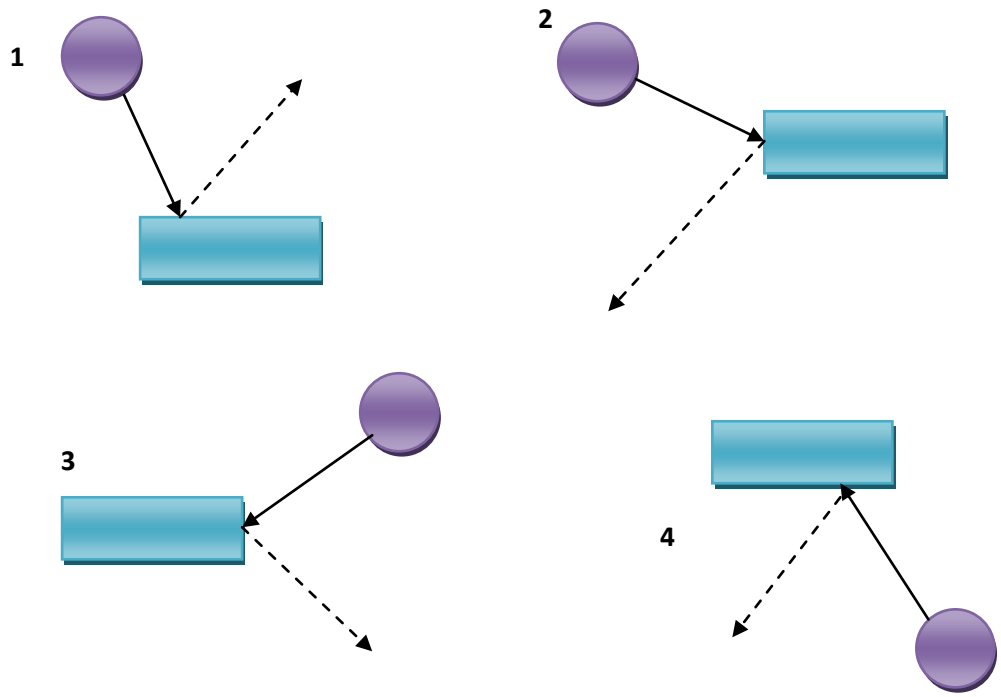
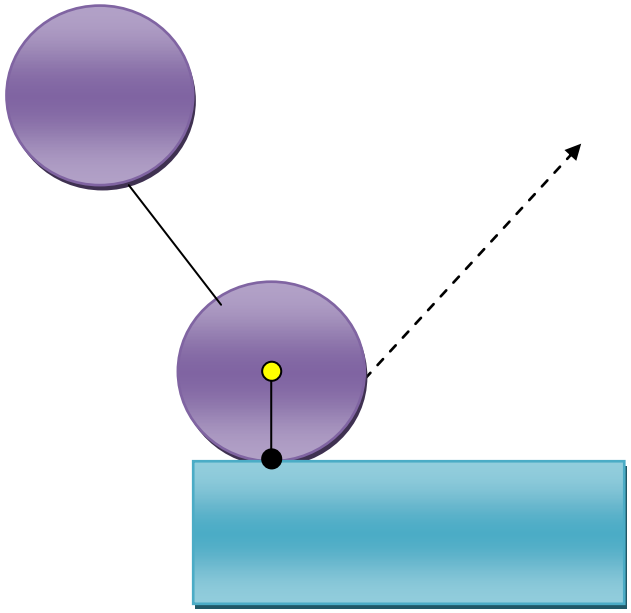
Ayuda

1. Colisión.

¿De cuantas maneras puede colisionar la pelota con un bloque?

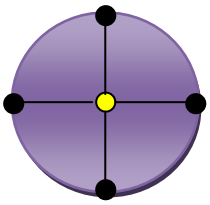


Veamos el caso 1 desde más cerca:



Básicamente tienen que detectar, si el punto negro se encuentra en el interior del cuadrado.

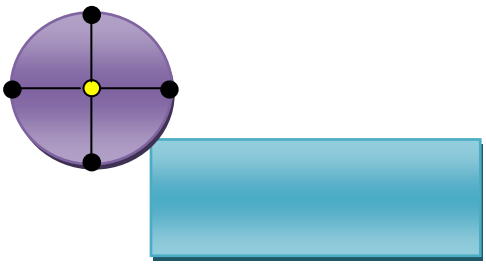
Aplicando la misma analogía a las otras colisiones solo tendrían que detectar si los siguientes 4 puntos se encuentran en el interior de un rectángulo.



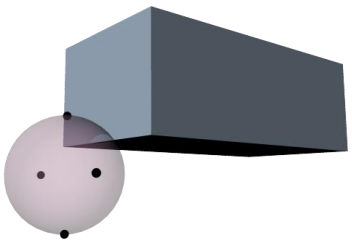
Estos 4 puntos son fáciles de calcular, ya que tienen como información de entrada el centro del círculo y el radio del mismo.

Aunque estos 4 casos se ejemplifican en un caso 2D, el análisis es igual para en el caso 3D.

Caso Especial



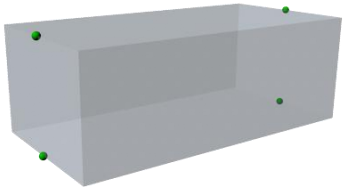
O su equivalente 3D a:

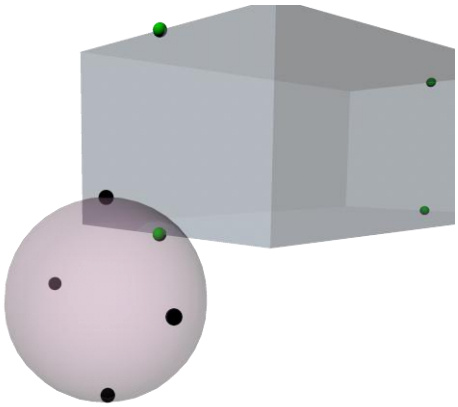


La técnica de arriba no se aplica a este caso, pueden ver que hay colisión pero ningún punto negro choca con el rectángulo.

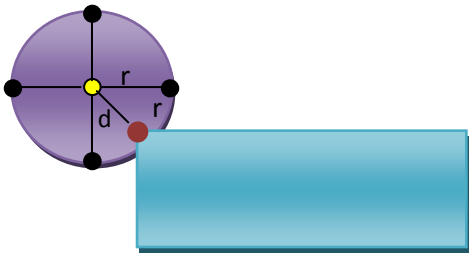
Solución:

En este caso lo que hacen es tomar los puntos medios de las aristas de las caras laterales del paralelepípedo:





Hacen el proceso inverso, verifican si los puntos medios del paralelepípedo se encuentran en el interior de la esfera.



Es resumen, para verificar la colisión entre un una esfera y un paralelepípedo tienen que emplear el siguiente procedimiento:

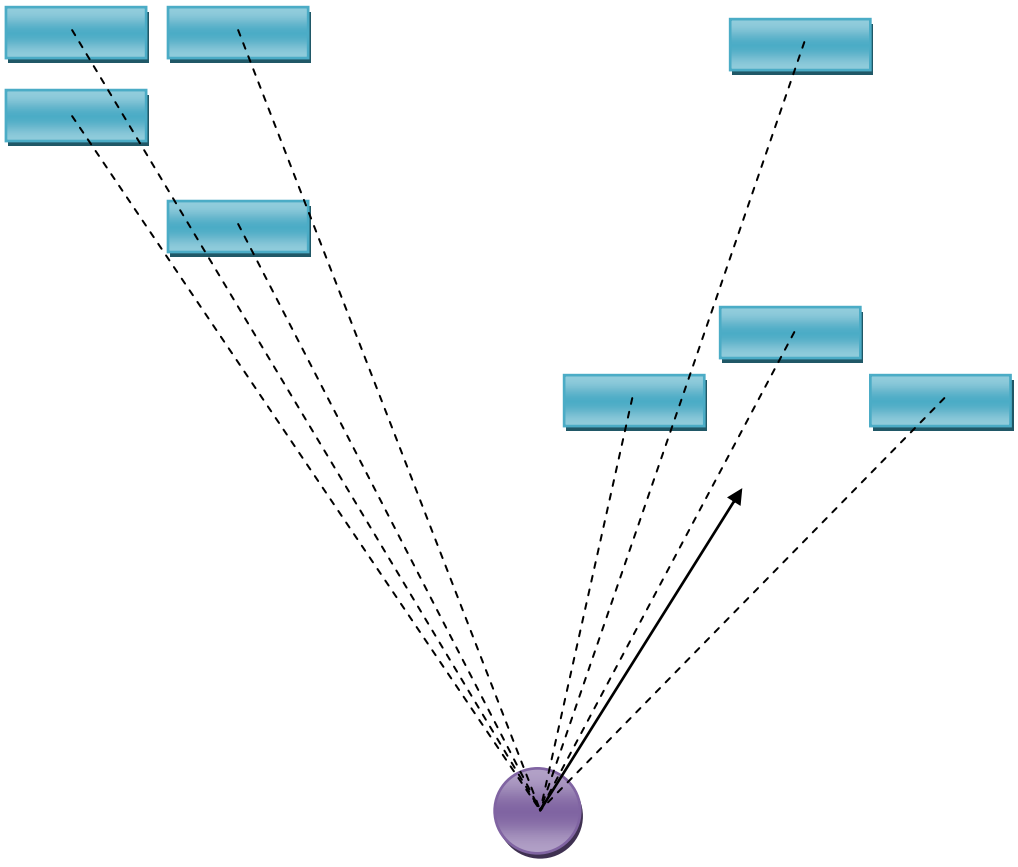
1. Verificar si algunos de los 4 puntos negros se encuentra en el interior del paralelepipedo.
2. Verificar si los los puntos medio de las caras laterales se encuentran en el interior de la esfera

Cabe mencionar que estos dos pasos lo pueden hacer en un solo ciclo, no es necesario generar 2 ciclos para verificar los pasos anteriores.

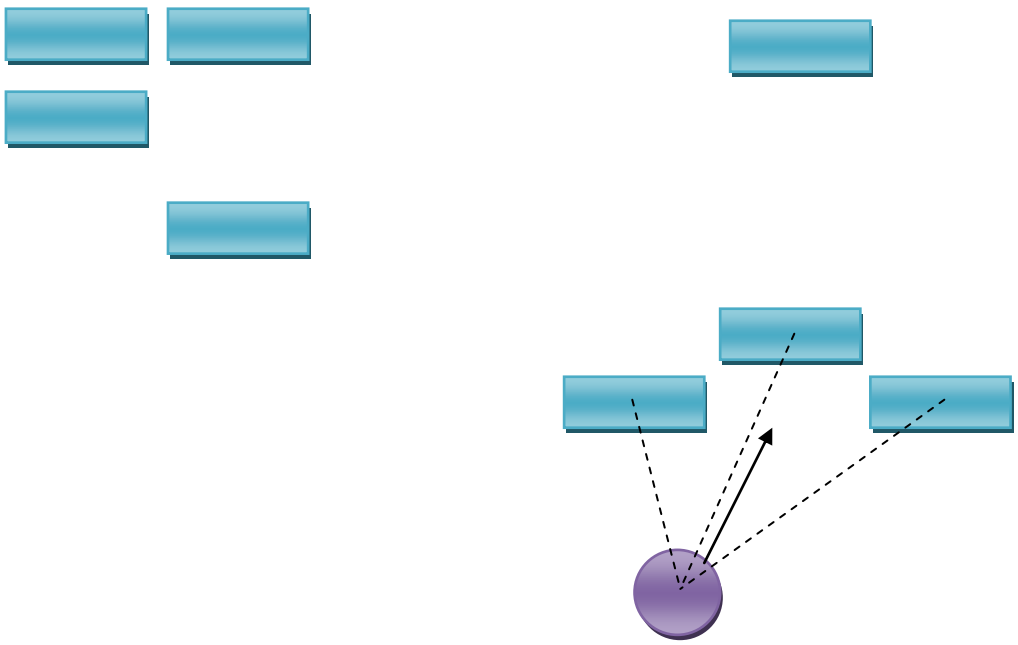
Nota Opcional (Eficiencia en el código)

Este consejo es opcional y no es obligado su implementación, según los pasos descritos arriba, cada vez que ustedes mandan a dibujar o mueven el circulo tienen que validar si los 4 puntos colisionan con un rectángulo, es decir, si hay 100 rectángulos, tienen que hacer la validación en cada uno de los rectángulos, lo cual hace el proceso un poco lento.

Aunque menciono “lento”, es importante recalcar que un ciclo de 100 operaciones es algo que una computadora actual puede hacer en tiempo real. Pero si desean optimizar su código pueden hacer lo siguiente:



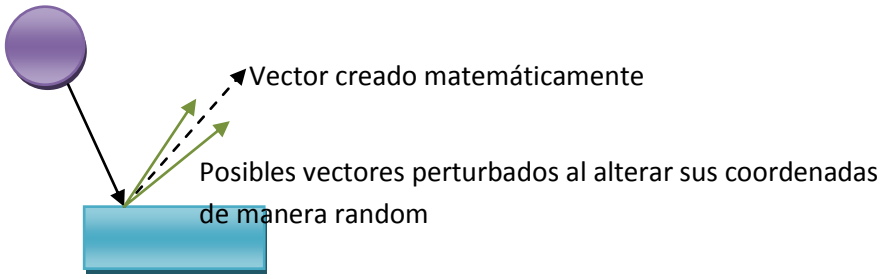
Se calcula la distancia que hay entre el circulo y cada rectángulo, tomando como referencia el centro del rectángulo, fijan una distancia “probable”, dicha distancia le dice al programa si el circulo esta cerca de algunos rectángulos. Si la distancia de los rectángulos al círculo es menor que la distancia “probable”, entonces proceden a validar la intersección entre el circulo y los rectángulos que están cerca de él.



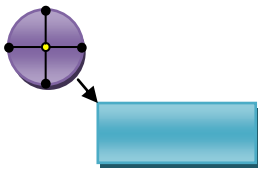
Siguiendo el ejemplo y usando esa técnica solamente se tiene que validar la intersección en 3 rectángulos y no en todos los rectángulos, esto optimiza considerablemente el tiempo de ejecución. Esta explicación se preserva para el espacio 3D

2. Reflexión.

Cuando apliquen la reflexión “perturben” el vector reflejado, ya que de lo contrario siempre la pelota se moverá de la misma manera cada vez que inicien su programa.



Caso especial:



Aquí se le deja al estudiante, la libertad y creatividad de elegir alguna solución para este caso