

# Portal 0.0

BADSTÜBER Elian BIDAULT Matthieu FOCHEUX Vital  
Licence 3 Informatique

Mars 2024



Tuteur : Julien BERNARD

# Table des matières

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

- Les rendus
- Les portails

## 4 Conclusion

# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

- Les rendus
- Les portails

## 4 Conclusion

# Introduction

## Système de jeu

- Portal 0.0 → principes techniques de plusieurs jeux vidéos connus
- Résolution d'énigmes à l'aide de portails
- Téléportation lorsqu'on passe à travers
- Principe de Portal (2007)



Figure: Portal (2007)

# Introduction

## Technique graphique

- Portal 0.0 → principes techniques de plusieurs jeux vidéos connus
- Méthode raycasting
- Rendu 2.5D popularisé dans les années 90
- Principe de Wolfenstein3D (1992)



Figure: Doom (1993)

# Introduction

## Technologies utilisées



# Plan

1 Introduction

2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

3 Implémentation

- Les rendus
- Les portails

4 Conclusion

# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

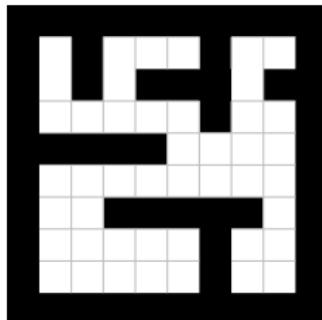
- Les rendus
- Les portails

## 4 Conclusion

# Construction de mur

## Commencement

- Création de la carte à partir d'un PNG
- Récupération des coordonnées des cellules à l'aide d'un parcours en profondeur

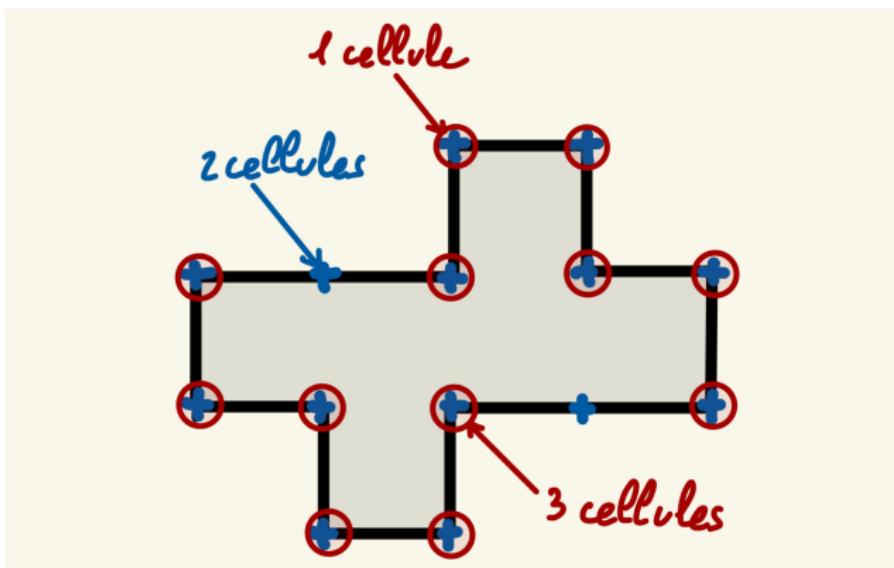


- Les coordonnées des cellules correspondent au coin supérieur gauche de chaque cellule
- Une cellule correspond à un bloc unitaire de la carte

# Construction de mur

## Sommets utiles

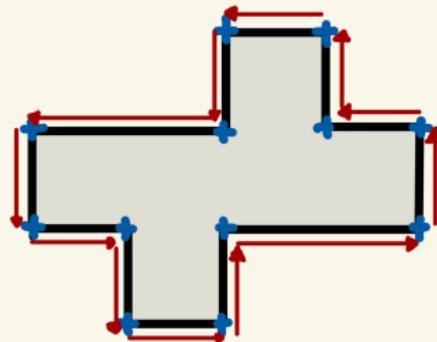
- Boucle sur les coordonnées des cellules
- Parité du nombre de cellules adjacentes



# Construction de mur

## Trie des sommets

```
d = {'d', 'b', 'g', 'h'}
dir = d[i%4], dir2 = d[(i+2)%4]
while(uv.contient(sommetA(dir)))
    ou uv.contient(sommetA(dir2))):
    if canGo(dir) and !canGo(dir2):
        v <- sommetA(dir)
    else:
        v <- sommetA(dir2)
    echange(dir, dir2)
```



# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- **Les collisions**
- Différentes stratégies pour un rendu 3D

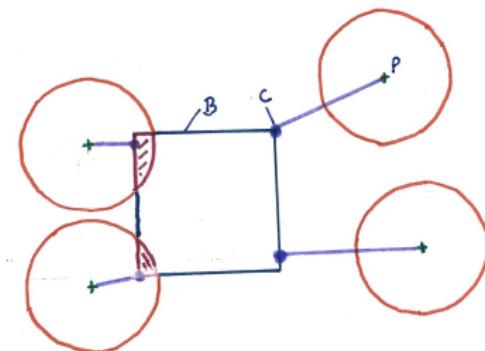
## 3 Implémentation

- Les rendus
- Les portails

## 4 Conclusion

# Les collisions

## Système de collision cercle/rectangle



Different cases of clamping.

- Recherche du point le plus proche
  - ▶ méthode de clampage
- Validation de proximité
  - ▶ replacement du personnage en cas de collision

# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

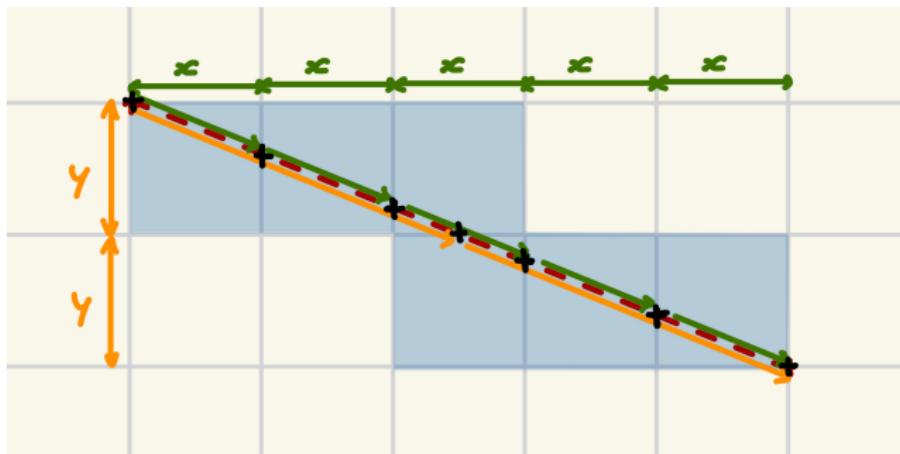
- Les rendus
- Les portails

## 4 Conclusion

# Différentes stratégies pour un rendu 3D

## DDA : l'algorithme commun

- Conçu pour la rasterisation de lignes
- Repose sur l'itération linéaire
- Employé pour déterminer où les rayons projetés intersectent avec les objets de l'environnement

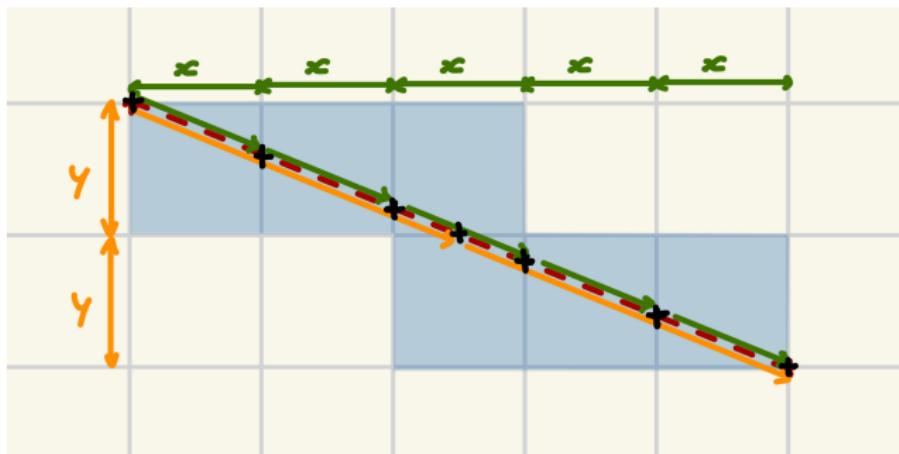


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

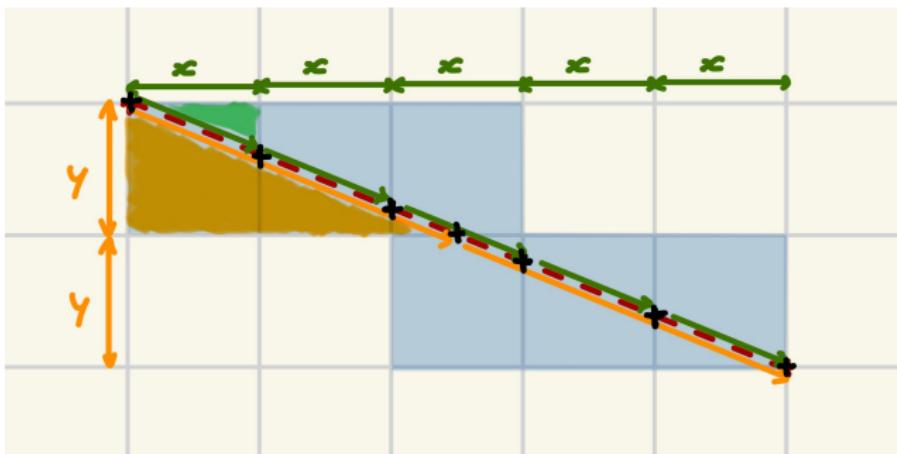


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

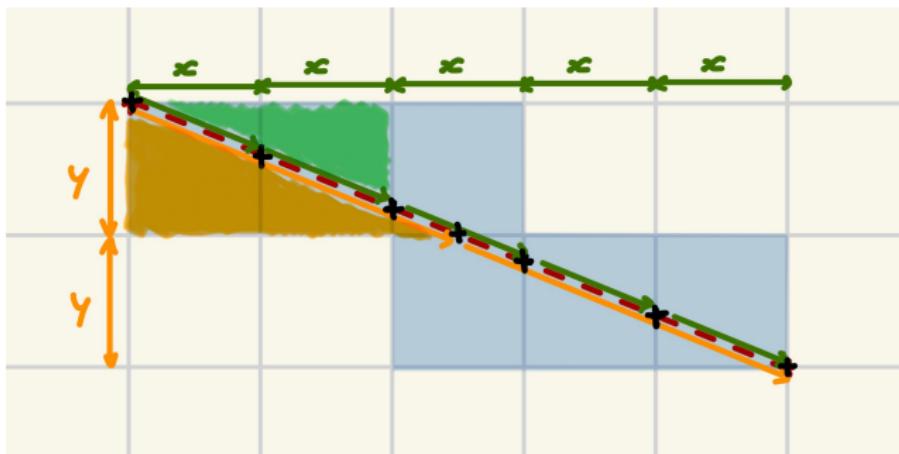


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

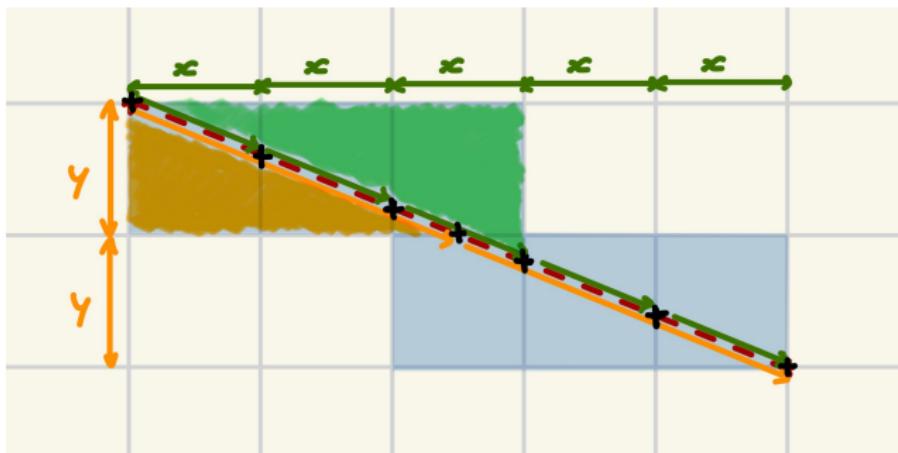


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

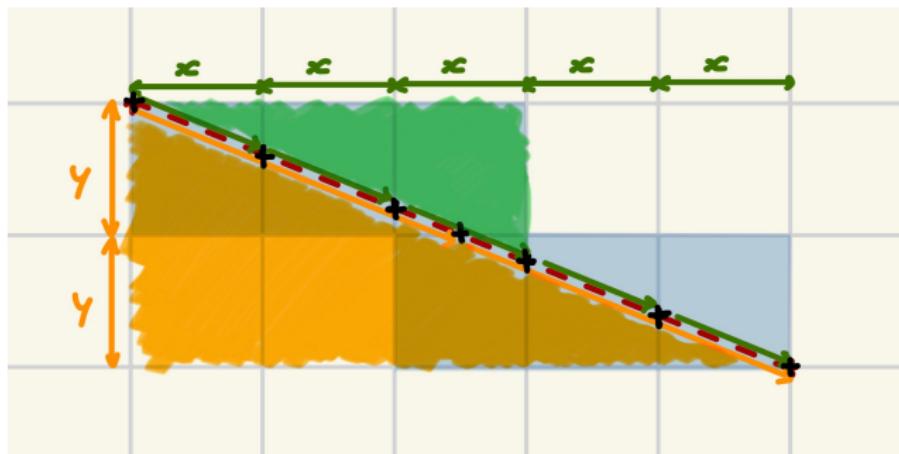


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

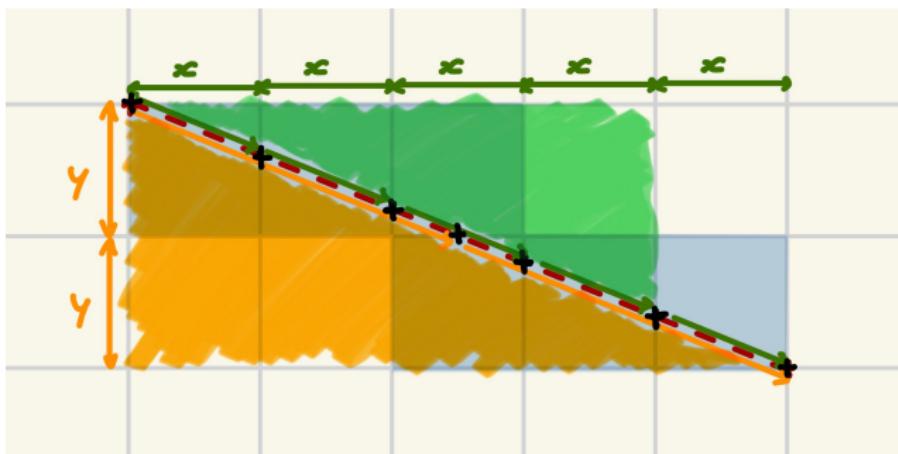


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

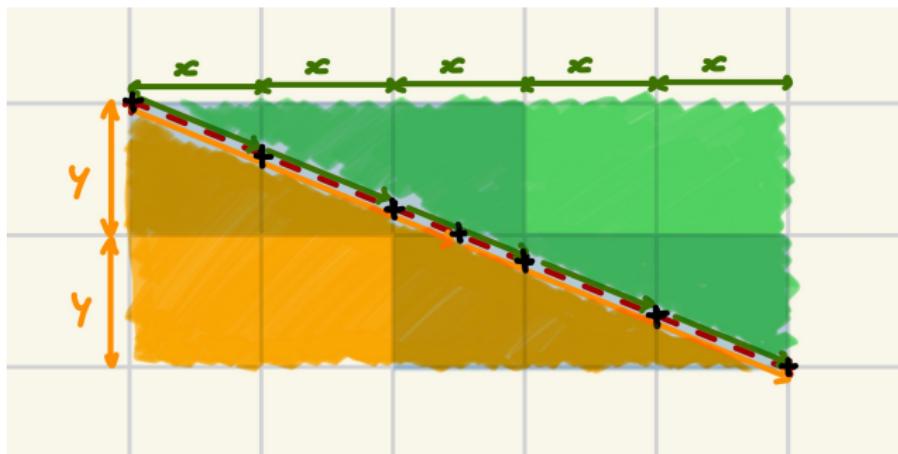


# Différentes stratégies pour un rendu 3D

DDA : l'algorithme commun

## Fonctionnement de DDA.

- Déplacement vertical et horizontal d'une unité.
- Comparaison entre les distances parcourus sur le segment.
- Choix de la distance la plus petite.

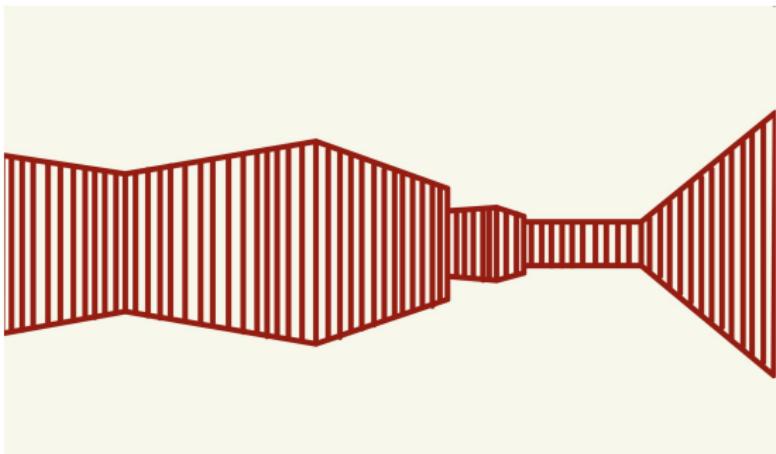
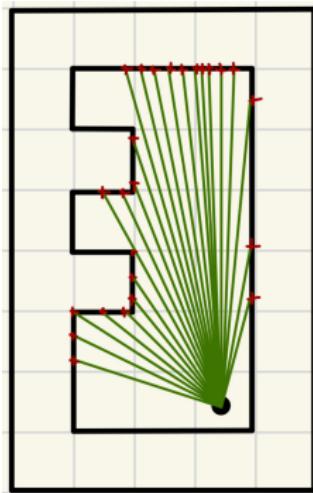


# Différentes stratégies pour un rendu 3D

## Approche historique

### Raycasting

- Projection de rayons depuis la position du joueur.
- Répétition pour chaque colonnes de pixels de la fenêtre.

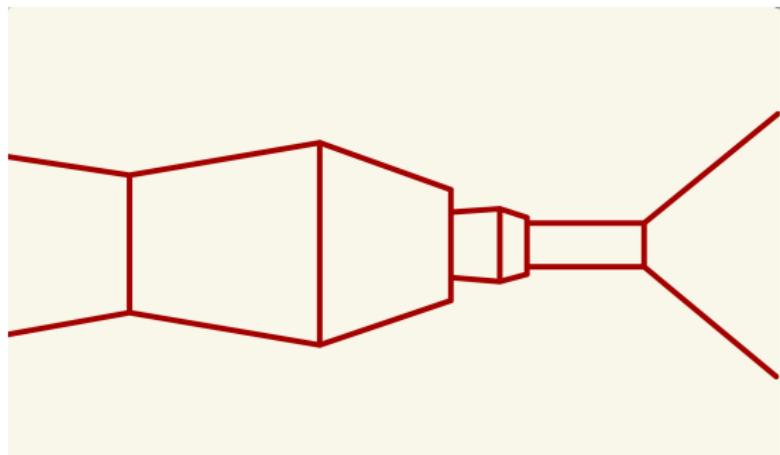
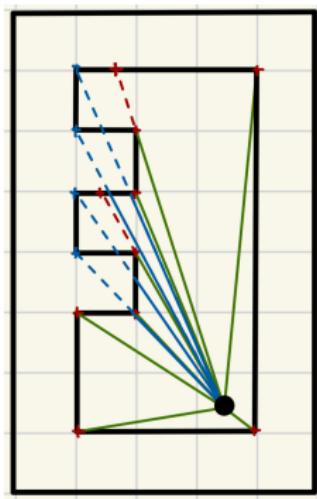


# Différentes stratégies pour un rendu 3D

## Approche moderne

### Line Of Sight

- Envoyer un rayon pour chaque sommet de chaque mur



# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

- Les rendus
- Les portails

## 4 Conclusion

# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

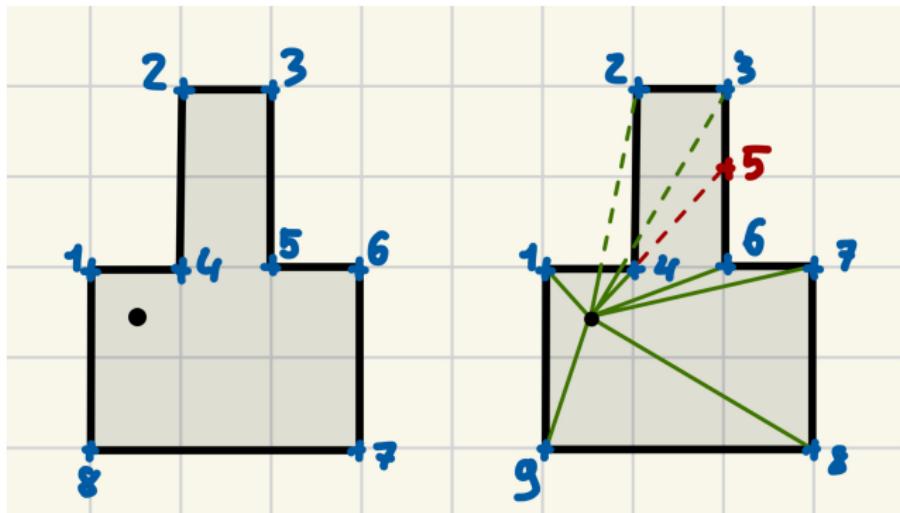
- Les rendus
- Les portails

## 4 Conclusion

# Les rendus

## Trois étapes

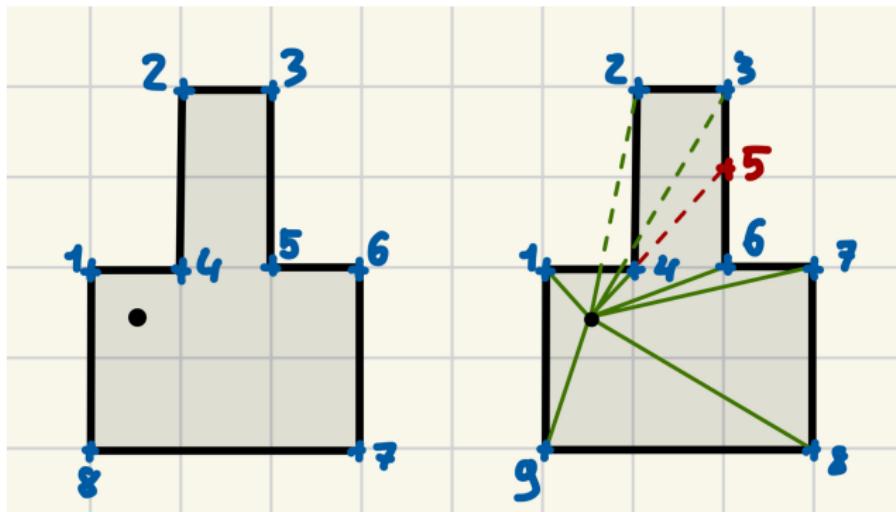
- Récupération des sommets triés
- Envoie des rayons dans l'ordre avec DDA
- Déterminer la position et la taille des segments sur la fenêtre de jeu



# Les rendus

## Trois cas possibles

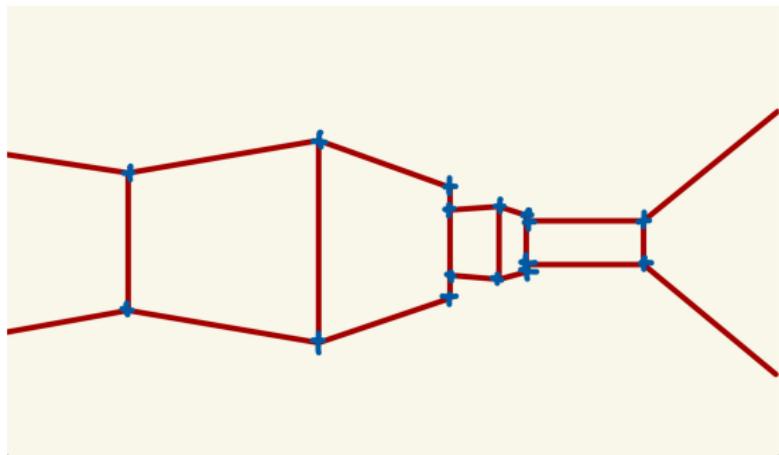
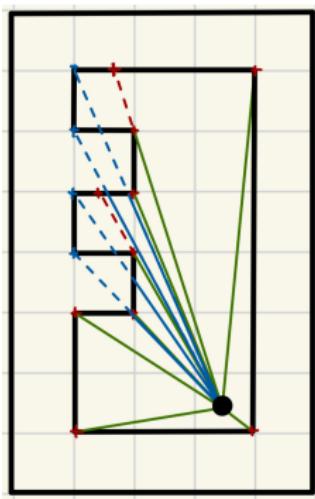
- Le sommet est atteint et on arrête le rayon.
- Le sommet n'est pas atteint donc on arrête le rayon.
- Le sommet est atteint mais on continue le rayon.



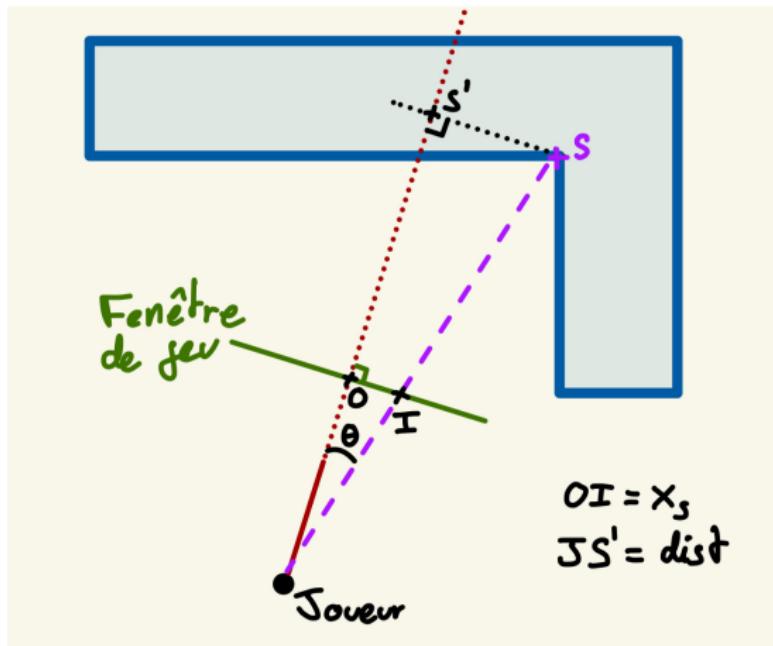
# Les rendus

## Réultat de la projection

- Angle du rayon → position du segment.
- Distance du rayon → hauteur du segment.



## Les rendus



## Projection des sommets

- $OI = \tan \theta$ , la position horizontale du segment.
- $h = \frac{H}{2dist}$ , la hauteur du segment.

# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

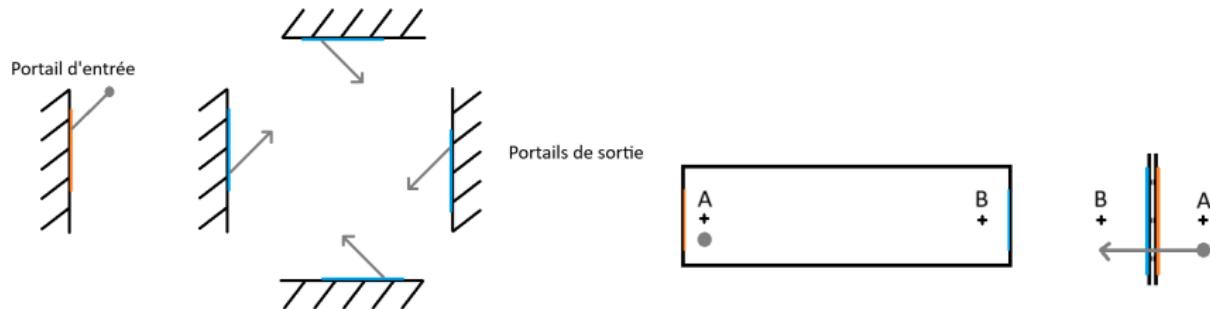
## 3 Implémentation

- Les rendus
- Les portails

## 4 Conclusion

# Les portails

## La téléportation



- Utilisation des cliques gauche et droit
  - Gain de distance
- 
- Réutilisation du système de collision
  - Rencontre avec un portail
    - ▶ Calcul du nouvelle angle
    - ▶ Calcul de la nouvelle position

# Plan

## 1 Introduction

## 2 Algorithmes et techniques

- Construction de mur
- Les collisions
- Différentes stratégies pour un rendu 3D

## 3 Implémentation

- Les rendus
- Les portails

## 4 Conclusion

# Conclusion

## Ce qui a été fait

- Création d'un environnement de jeu
  - ▶ Grille de jeu
  - ▶ Système de collisions
  - ▶ Rendu texturé
  - ▶ Portails
- Mise en place du raycaster
  - ▶ Utilisation de l'algorithme DDA

# Conclusion

## Améliorations possible

- Utilisation de scènes
- Vue à travers le portail
- Ajout de niveaux supplémentaires
- Ajout d'un marquage de fin de niveau

# Questions

Merci de votre attention.  
Avez-vous des questions ?