

RÉALITÉ AUGMENTÉE (RA)

L'objectif pédagogique du cours est de permettre aux apprenants de comprendre et de mettre en œuvre une application de réalité augmentée intégrant plusieurs compétences clés : détection robuste en vision par ordinateur d'un marqueur rectangulaire (feuille A4), estimation de la pose 3D du marqueur en temps réel, et superposition en 3D d'un objet virtuel en cohérence avec ce marqueur. Le travail demandé vise à développer des savoir-faire pratiques en calibration de caméra, traitement d'image, algorithmie de suivi et rendu 3D via OpenGL, tout en sensibilisant à la gestion des ressources, à la structuration modulaire du code, et à l'optimisation des performances pour un usage temps réel. Il s'inscrit dans une démarche pédagogique favorisant l'autonomie, l'expérimentation et la validation sur des données réelles et variées (image fixe, séquences vidéo, flux caméra direct). L'évaluation de ce travail sera guidée par la grille suivante :

1. ARCHITECTURE ET ORGANISATION DU CODE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|------------------------|-------------------------------------|--|--|---|
| Structure modulaire | Code monolithique sans organisation | Quelques fonctions, organisation basique | Structures de données dédiées, fonctions bien organisées | Architecture claire avec « <i>namespaces</i> », structures cohérentes ... |
| Documentation | Aucune documentation | Commentaires sporadiques | Commentaires sur fonctions principales | Documentation « <i>Doxygen</i> » complète avec descriptions détaillées |
| Gestion des ressources | Fuites mémoire, pas de nettoyage | Nettoyage partiel | Gestion correcte des ressources OpenGL/OpenCV | Nettoyage systématique avec fonctions dédiées |

2. VISION PAR ORDINATEUR (OPENCV)

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|-----------------------|-------------------------------------|--|---|---|
| Détection de marqueur | Détection basique non fonctionnelle | Détection simple sans robustesse | Détection avec seuillage adaptatif | Détection robuste avec ROI adaptative et validation temporelle |
| Calibration caméra | Pas de calibration | Calibration manuelle basique | Chargement paramètres depuis fichier | Gestion complète matrice intrinsèque et coefficients distorsion |
| Estimation de pose | Pas d'estimation 3D | Utilisation de la fonction <i>solvePnP</i> basique sans validation | Estimation avec points triés correctement | Estimation robuste avec validation cohérence inter-frames |
| Traitement d'image | Opérations élémentaires | Conversion couleur et seuillage | Pipeline complet avec filtrage | Pipeline optimisé avec adaptation dynamique des paramètres |

3. INTÉGRATION AR ET SUIVI

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|------------------|-----------------------------|---------------------------|-------------------------------------|---|
| Robustesse suivi | Perte fréquente du marqueur | Suivi instable avec sauts | Suivi stable en conditions normales | Validation temporelle avec seuil de cohérence, gestion de l'éclairage |
| Superposition 3D | Alignement incorrect | Alignement approximatif | Bonne superposition | Superposition précise avec perspective correcte et repères visuels |

| | | | | |
|------------------------|------------------------------|--------------------------------|-------------------------------|--|
| Performance temps réel | < 15 fps, latence importante | 15-20 fps, latence visible | 20-25 fps, latence acceptable | ≥ 25 fps stable avec pipeline optimisé |
| Enregistrement vidéo | Pas d'enregistrement | Capture basique avec artéfacts | Enregistrement fonctionnel | Capture via la fonction <i>glReadPixels</i> avec conversion et redimensionnement automatique |

4. QUALITÉ TECHNIQUE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|---------------------|-----------------------------------|-------------------------------------|----------------------------------|--|
| Gestion d'erreurs | Pas de gestion, crashes fréquents | Try-catch basique | Gestion avec messages explicites | Validation systématique avec exceptions typées et messages détaillés |
| Configuration | Valeurs en dur dans le code | Quelques constantes | <i>Namespace</i> Config dédié | Configuration complète centralisée et documentée |
| Complexité visuelle | Objet 3D simple | Cube + axes | Cube + axes + marqueurs | Rendu multiple avec cercles aux coins, codes couleur cohérents, etc... |
| Adaptabilité | Code non modifiable | Paramètres modifiables manuellement | Paramètres centralisés | Architecture extensible pour nouveaux marqueurs/objets 3D |

5. ALGORITHMES ET TECHNIQUES AVANCÉES

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|----------------------------|-----------------------------------|-----------------------------|-------------------------------------|--|
| Tri de points | Ordre incorrect ou aléatoire | Tri basique par coordonnées | Algorithme avec séparation haut/bas | Tri robuste garantissant correspondance coins physiques/virtuels |
| Transformation coordonnées | Conversion incorrecte | Conversion partielle | Matrices complètes OpenCV -> OpenGL | Pipeline complet avec Rodrigues, transposition et conversion types |
| Optimisation pipeline | Calculs redondants à chaque frame | Quelques optimisations | Pipeline structuré | Réutilisation textures, mise à jour conditionnelle, buffering efficace |

6. TESTS ET VALIDATION SUR DONNÉES RÉELLES

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|---------------------------|---------------------------------------|---|--|--|
| Image statique fournie | Ne fonctionne pas sur l'image test | Détection partielle avec erreurs | Détection fonctionnelle avec quelques imprécisions | Détection et superposition 3D parfaitement alignées |
| Séquence vidéo fournie | Ne fonctionne pas sur la vidéo test | Suivi instable, nombreuses pertes | Suivi fonctionnel avec pertes occasionnelles | Suivi robuste sans perte sur toute la séquence |
| Vidéo acquise par l'élève | Pas de vidéo personnelle testée | Ne fonctionne que dans des conditions idéales | Fonctionne dans des conditions variées | Fonctionne avec différents éclairages, angles et distances |
| Flux caméra temps réel | Pas d'implémentation caméra en direct | Fonctionne mais avec latence importante | Suivi temps réel fonctionnel | Suivi temps réel fluide et robuste avec adaptation automatique |

SYSTÈME DE NOTATION

Échelle d'évaluation :

- Insuffisant (1 point) : Compétences de base non maîtrisées
- Satisfaisant (2 points) : Compétences de base acquises
- Bon (3 points) : Compétences intermédiaires maîtrisées
- Excellent (4 points) : Compétences avancées maîtrisées

Barème indicatif :

- 23-39 points : Niveau débutant - Compétences à développer
- 40-59 points : Niveau intermédiaire - Bases solides

- 60-79 points : Niveau avancé - Bonne maîtrise
- 80-88 points : Niveau expert - Excellente maîtrise

Note importante sur la note finale :

- La subsection "Tests et Validation sur Données Réelles" est essentielle pour valider la robustesse de l'application. Un projet excellent doit fonctionner sur différents types de données (image fixe, vidéo préenregistrée, vidéo personnelle et flux caméra temps réel).
- L'expérience utilisateur sera prise en compte pour la note finale.

La note finale sera pondérée au regard de ces deux points.

RÉALITÉ VIRTUELLE (RV)

L'objectif pédagogique du cours est de permettre aux apprenants de comprendre et de mettre en œuvre une application de réalité virtuelle. Le travail demandé vient en continuité du cours de réalité augmentée en :

- Réutilisant les compétences de suivi (tracking) et calibration
- Étendant la complexité des scènes 3D (de cube simple à environnement)
- Approfondissant OpenGL (éclairage, texturage, ombres)
- Explorant les modes de représentation (vidéo vs environnement virtuel)
- Consolidant la gestion temps réel et l'optimisation

L'apprenant sera capable de concevoir et d'assembler une architecture logicielle cohérente qui réutilise un code AR existant et y ajoute le labyrinthe, la sphère texturée, l'ombre et la gestion des deux modes d'affichage. Il pourra créer une mise en scène 3D complète autour du marqueur (organisation du labyrinthe, placement de la lumière, choix du fond virtuel) en tenant compte à la fois des contraintes techniques et de la lisibilité pour l'utilisateur.

1. ARCHITECTURE ET ORGANISATION DU CODE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|-----------------------------|-------------------------------------|--|--|---|
| Structure modulaire | Code monolithique sans organisation | Quelques fonctions, organisation basique | Structures de données dédiées, fonctions bien organisées | Architecture claire avec « <i>namespaces</i> », structures cohérentes ... |
| Documentation | Aucune documentation | Commentaires sporadiques | Commentaires sur fonctions principales | Documentation « <i>Doxygen</i> » complète avec descriptions détaillées |
| Gestion des ressources | Fuites mémoire, pas de nettoyage | Nettoyage partiel | Gestion correcte des ressources OpenGL/OpenCV | Nettoyage systématique avec fonctions dédiées |
| Intégrité du code hérité RA | Le prototype RA est cassé | Le prototype RA fonctionne mais modifié | Le prototype RA conservé avec modifications minimales | Prototype RA complètement préservé, évolution clean vers VR sans régression |

2. GÉOMÉTRIE ET CONSTRUCTION DU LABYRINTHE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|------------------------------------|--|---------------------------------|--|--|
| Positionnement relatif au marqueur | Murs mal placés, non alignés | Murs grossièrement positionnés | Murs bien positionnés autour du marqueur | Disposition précise avec marges cohérentes, structure géométrique robuste |
| Dynamique de suivi | Labyrinthe figé, ne suit pas le marqueur | Suivi imprécis ou avec décalage | Suivi correct du marqueur | Suivi fluide et précis, adaptation automatique de la matrice de transformation |

3. SPHÈRE TEXTURÉE ET MATÉRIAUX

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|-------------------------|------------------------|--------------------------------------|--------------------------------|--|
| Génération de la sphère | Absente | Sphère basique sans qualité visuelle | Sphère avec UV mapping correct | Sphère optimisée avec paramétrage UV précis |
| Texturage | Pas de texture | Texture appliquée (UV incorrectes) | Texture correcte | Texture de haute qualité (mipmapping, filtrage ...) |
| Propriétés matériau | Pas de matériau défini | Couleur basique uniquement | Diffuse + specular | Diffuse + specular + ambient + shininess paramétrables |

4. OMBRES ET OMBRAGE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|-------------------|-----------------------------|--|--------------------------------------|--------------------------------|
| Ombre simple | Pas d'ombre | Ombre basique (projection orthogonale) | Ombre au sol avec géométrie correcte | Ombre avec perspective adaptée |
| Performance ombre | Ralentissement significatif | Léger ralentissement | Impact négligeable (< 5% fps) | Aucun impact ressenti |

5. ÉCLAIRAGE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|---------------------|-------------------------------|---------------------------------------|---|--|
| Modèle d'éclairage | Pas d'éclairage, flat color | Éclairage diffus basique (Lambertien) | Phong ou Blinn-Phong complet | PBR (Physically Based Rendering) ou Phong avancé |
| Shaders spécialisés | Shaders génériques ou absents | 1 seul shader pour tout | Shaders séparés pour labyrinthe et sphère | Shaders optimisés par type de géométrie |

6. GESTION DES MODES AR/VR

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|---------------------|-----------------------------|------------------------------------|---|--|
| Mode Augmenté (AR) | Absent ou non fonctionnel | Affichage vidéo + 3D avec décalage | Superposition correcte vidéo et 3D | Superposition précise avec gestion profondeur, blending adapté |
| Mode Virtuel (VR) | Absent ou non implémentable | Basculement basique, fond blanc | Basculement vers fond virtuel fonctionnel | Fond virtuel (skybox, couleur, texture) avec transition fluide |
| Basculement AR ↔ VR | Non implémenté | Basculement manuel fragile | Basculement stable via touche | Basculement instantané et robuste, gestion des ressources propre |

7. RENDU 3D ET PERFORMANCE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|--------------------------|-----------------------------------|--------------------------------|---|--|
| Qualité visuelle globale | Aliasing visible, textures floues | Qualité acceptable en standard | Bonne qualité visuelle, peu d'artefacts | Rendu lisse, sans aliasing notable, cohérence visuelle |
| Performance temps réel | < 15 fps instable | 15-20 fps avec variations | 20-25 fps stable | ≥ 25 fps constant avec headroom (< 80% charge) |

SYSTÈME DE NOTATION

Échelle d'évaluation :

- Insuffisant (1 point) : Compétences de base non maîtrisées
- Satisfaisant (2 points) : Compétences de base acquises
- Bon (3 points) : Compétences intermédiaires maîtrisées
- Excellent (4 points) : Compétences avancées maîtrisées

Barème indicatif :

- 20-33 points : Niveau débutant - Compétences à développer
- 34-46 points : Niveau intermédiaire - Bases solides
- 47-59 points : Niveau avancé - Bonne maîtrise
- 60-72 points : Niveau expert - Excellente maîtrise

L'expérience utilisateur sera prise en compte pour la note finale.

RÉALITÉ MIXTE (RM)

L'objectif pédagogique du cours est de permettre aux apprenants de concevoir une expérience de réalité mixte interactive, où des objets virtuels coexistent et interagissent avec un plan réel tracké en temps réel. Ils devront implémenter une bille dont le mouvement dépend directement de l'inclinaison du plan.

Pour intégrer convenablement la communication homme-machine, Il s'agira également de mettre en œuvre une physique de jeu robuste : collisions bille-murs, rebonds, stabilité numérique et jouabilité en conditions réelles. Les apprenants développeront une génération procédurale de labyrinthes à complexité contrôlée, avec un départ et une arrivée garantissant une progression cohérente. Enfin, ils mettront en place une boucle de jeu (démarrer, valider la sortie, passer au niveau suivant) et, en option, des paramètres ajustables (vitesse, restitution des rebonds) pour expérimenter l'équilibrage.

1. ARCHITECTURE ET ORGANISATION DU CODE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|------------------------|-------------------------------------|--|--|---|
| Structure modulaire | Code monolithique sans organisation | Quelques fonctions, organisation basique | Structures de données dédiées, fonctions bien organisées | Architecture claire avec « <i>namespaces</i> », structures cohérentes ... |
| Documentation | Aucune documentation | Commentaires sporadiques | Commentaires sur fonctions principales | Documentation « <i>Doxygen</i> » complète avec descriptions détaillées |
| Gestion des ressources | Fuites mémoire, pas de nettoyage | Nettoyage partiel | Gestion correcte des ressources OpenGL/OpenCV | Nettoyage systématique avec fonctions dédiées |

2. PHYSIQUE BILLE ET PLAN

Le mouvement attendu peut être modélisé en projetant la gravité sur le plan incliné et en intégrant la dynamique à chaque frame.

<https://www.physicsclassroom.com/class/vectors/Lesson-3/Inclined-Planes>

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|--------------------------------|---|--|---|--|
| Mouvement selon inclinaison | Mouvement absent ou incohérent | Mouvement présent mais peu fidèle (sens/échelle discutables) | Mouvement cohérent (direction, échelle), stable. | Mouvement réaliste avec paramétrage (gain, friction), stabilité exemplaire |
| Conversion pose → inclinaison | Ne tient pas compte de l'orientation trackée. | Prend en compte une partie des axes, erreurs fréquentes | Exploite correctement l'orientation du plan pour piloter la bille | Mapping robuste (axes, repères), indépendant des conventions caméra/monde |
| Intégration temporelle | Instable (explosions, oscillations) | Stable mais dépend fortement du FPS | Stable avec dt explicite et limites raisonnables | Stable et robuste (clamp dt , semi-implicite, comportement identique selon FPS) |
| Dissipation (friction) | Pas de friction, bille "glisse" indéfiniment | Friction ad-hoc, résultats peu contrôlables | Friction cohérente, paramétrable | Pas de friction, bille "glisse" indéfiniment |
| Stabilité avec tracking bruité | La bille devient inutilisable (sauts) | Filtrage minimal, "jitter" visible | Filtrage/atténuation efficace (lissage pose ou accélération) | Compensation avancée (filtrage temporel, anti-sauts), très jouable en conditions réelles |

3. COLLISIONS ET REBONDS

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|--------------------------------|-------------------------------------|--|--|--|
| Détection sphère-murs | Aucune ou très incomplète | Détection partielle, collisions ratées | Détection fiable sur l'ensemble des murs | Détection robuste (coins, couloirs étroits), sans "tunneling" notable |
| Réponse au contact | Bille traverse ou colle aux murs | Rebond présent mais instable | Rebond stable avec restitution correcte | Rebond stable + gestion frottement tangentielle, restitution paramétrable |
| Confinement dans le labyrinthe | Sort de la zone/monde. | Confinement approximatif | Confinement correct et constant. | Confinement impeccable, y compris lors de tracking partiel/latence |
| Gestion des cas limites | Comportements erratiques aux angles | Coin/angles parfois problématiques. | Angles gérés correctement | Gestion avancée (résolution itérative, correction pénétration), très robuste |

4. GÉNÉRATION AUTOMATIQUE DU LABYRINTHE

Les algorithmes "classiques" de génération (ex. DFS/backtracking, Prim) permettent de générer des labyrinthes connectés, souvent "parfaits" (un seul chemin entre deux cellules), ce qui facilite la maîtrise de la difficulté.

<https://dumbgenius.github.io/experiments/mazegen/>

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|-----------------------------|---|--|---|---|
| Génération procédurale | Labyrinthe figé (manuel) ou non fonctionnel | Génération simple mais peu contrôlée | Génération fonctionnelle avec paramètres de taille | Génération maîtrisée (taille + complexité) et répétable (seed) |
| Maîtrise de la complexité | Impossible d'ajuster la difficulté | Ajustement grossier (taille seulement) | Ajustement via plusieurs paramètres (taille, densité, couloirs) | Ajustement fin + progression par niveaux (courbe de difficulté) |
| Départ & arrivée (validité) | Départ/arrivée absents ou non atteignables | Parfois atteignables | Toujours atteignables, règles claires | Validation automatique (chemin garanti), placement pertinent et équilibré |
| Génération géométrie 3D | Murs mal générés/artefacts | Géométrie minimale mais cohérente | Géométrie correcte (murs, sol, bords) | Géométrie optimisée + cohérence visuelle (UV, normales), adaptée au jeu |

5. GAMEPLAY ET INTERFACE

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|---------------------------|-----------------------|--------------------------------|--|---|
| Boucle de jeu (états) | Pas de logique de jeu | Démarrage/fin partiels | États clairs (menu/démarrage, jeu, victoire) | Gestion complète (pause, restart, niveau suivant), transitions robustes |
| Condition de victoire | Non définie. | Définie mais peu fiable | Détection fiable de l'arrivée | Détection robuste + feedback clair (zone, effet, son/texte) |
| Passage au niveau suivant | Non implémenté | Implémenté de manière fragile. | Implémenté et stable | Progression fluide + difficulté qui augmente de façon maîtrisée |
| UI / HUD | Absent | UI minimale (texte brut) | UI claire (niveau, instructions, état) | UI soignée + options (réglages) + retour utilisateur (messages/indicateurs) |
| Paramètres optionnels | Non disponible | 1 paramètre modifiable en dur | Paramètres modifiables (vitesse, rebond...) | Paramètres modifiables + sauvegarde/presets + limites cohérentes |

4. TESTS

| Critère | Insuffisant (1) | Satisfaisant (2) | Bon (3) | Excellent (4) |
|-----------------------------|--------------------------------------|------------------------------------|--|---|
| Intégration MR (tracking) | Le monde 3D ne suit plus le marqueur | Suit partiellement, dérive notable | Suit correctement, stable | Très stable (filtrage, perte temporaire gérée), jouable en conditions réelles |
| Performance temps réel | Très lent / instable | Jouable mais chutes fréquentes | Fluide la plupart du temps | Fluide et constant, marge de performance |
| Robustesse (perte tracking) | Crash ou comportement incohérent | Se dégrade fortement | Dégradation acceptable, reprise correcte | Reprise propre (freeze, reset contrôlé), expérience utilisateur solide |
| Tests sur données | Aucun protocole de test | Test sur un seul cas | Test sur image/vidéo/flux caméra | Test complet + cas difficiles (éclairage, angles, occlusion partielle) |

SYSTÈME DE NOTATION

Échelle d'évaluation :

- Insuffisant (1 point) : Compétences de base non maîtrisées
- Satisfaisant (2 points) : Compétences de base acquises
- Bon (3 points) : Compétences intermédiaires maîtrisées
- Excellent (4 points) : Compétences avancées maîtrisées

Barème indicatif :

- 20-50 points : Niveau débutant - Compétences à développer
- 50-69 points : Niveau intermédiaire - Bases solides
- 70-84 points : Niveau avancé - Bonne maîtrise
- 85-100 points : Niveau expert - Excellente maîtrise

L'expérience utilisateur sera prise en compte pour la note finale.