Solution de géolocalisation dans le casque de Réalité Mixte HoloLens 2

Auteurs :

Bailly Éric (Eric.BAILLY@univ-cotedazur.fr)
Maître de Conférences, CMMC, Unité Propre de Recherche 1193, Université Nice Cote d'Azur.
Disciplines : Géographie – Aménagement
Thèmes de recherche : Modélisation, Simulation, Géovisualisation, Complexité, Fractals, Chaos, Dynamiques spatiales

Wahl Julien (julien.wahl@etu.univ-cotedazur.fr)
Étudiant, Département de Géographie et Aménagement, Université Nice Cote d'Azur.
Disciplines : Géographie – Aménagement
Thèmes de recherche : Géographie, Développement, Aménagement, SIG, Géovisualisation

Résumé :
La géographie analyse l'espace pour comprendre les lieux et les aménager. Elle emploie les techniques de géovisualisation pour les représenter. Les hologrammes, notamment via le casque HoloLens 2 de Microsoft, offrent des possibilités innovantes pour visualiser des objets géographiques en 3D. Cependant, ce casque manque de capacité de géoréférencement (GNSS), ce qui limite son usage en extérieur. Cet article expose une solution pour intégrer cette fonctionnalité à partir d'un smartphone. Les coordonnées de l'appareil, situé à côté du casque, lui sont envoyées, ajoutant ainsi une localisation précise des objets. L'application développée permet de récupérer et transmettre ces données au casque. La méthode utilise une programmation HTML pour obtenir les coordonnées et un serveur pour les synchroniser avec un ou plusieurs casques. Les tests montrent l'efficacité de cette approche.

Abstract: Geography analyzes space to understand and manage places, using geovisualization techniques to represent them. Holograms, particularly through Microsoft's HoloLens 2 headset, offer innovative possibilities for visualizing geographic objects in 3D. However, this headset lacks georeferencing capability (GNSS), limiting its outdoor use. This article presents a solution to integrate this functionality via a smartphone. The coordinates from the device, positioned next to the headset, are sent to it, thus adding precise location data to objects. The developed application retrieves and transmits these data to the headset. The method employs HTML programming to obtain coordinates and a server to synchronize them with one or multiple headsets. Tests demonstrate the effectiveness of this approach.

Mots clés :
Géovisualisation, Réalité mixte, GNSS, HoloLens 2, Géographie/Aménagement

Introduction

La géographie, en tant que science de l'aménagement et de l'analyse de l'espace, cherche à saisir la complexité des lieux qui composent notre monde afin d'agir dessus. L'espace est considéré comme son paradigme central. Son étude nécessite des outils et des méthodes capables de capturer, de visualiser et de comprendre sa complexité (Brunet, Ferras, Théry, 1992).

La géovisualisation, c'est-à-dire l'affichage des objets d'étude du géographe, est un important axe de recherche (Le Monde, 2019). Les pratiques de visualisation ont considérablement évolué, passant de maquettes en polystyrène à des technologies informatiques avancées. Aujourd'hui, l'utilisation des hologrammes offre de nouvelles perspectives pour représenter et interagir avec les données spatiales (Mericskay, 2022).

Une composante essentielle de la pertinence dans l'étude géographique est le géoréférencement, c'est-à-dire la possibilité de localiser avec exactitude un objet ou un phénomène sur terre dans un référentiel (par exemple la latitude et longitude). Cette capacité est déterminante non seulement pour la cartographie traditionnelle, mais aussi pour les applications modernes dans lesquelles la visualisation d'objets en 3D nécessite une précision spatiale rigoureuse (Géoconfluences).

Dans ce contexte, l'usage des hologrammes, matérialisés par le casque HoloLens 2 de Microsoft, introduit une innovation significative en matière de visualisation et d'interaction avec les objets géographiques (Wijesooriya, Farjad, Stergiou, Mastorakis, 2023). Toutefois, malgré ses avancées, l'HoloLens 2 présente une limitation majeure : l'absence de puce GNSS (global navigation satellite systems : GPS ou autres). Ainsi il n'est pas possible de géolocaliser un objet affiché. Cette carence pose un défi conséquent pour les applications nécessitant une localisation conforme et fiable (Rzeszewski, Orylski, 2021).

Cet article propose une méthode innovante pour intégrer la fonctionnalité de géoréférencement au casque HoloLens 2, ouvrant la voie à de nombreux usages liés aux hologrammes en géographie.

1 Hologrammes et potentialités en géographie et aménagement

Inventés en 1947 par le physicien hongrois Dennis Gabor, les hologrammes sont des images tridimensionnelles d'un objet, donc visibles sous tous les angles, apparaissant en suspension dans l'espace.
Bien qu'elles demeurent principalement des sujets de recherche, elles ouvrent de nouvelles perspectives pour les professionnels du secteur (Martin et al. 2019).

1.1 De la réalité virtuelle (RV) à la réalité mixte (RM)

L'évolution de la représentation des objets par l'informatique a conduit, dans les années 1990, au développement des premiers casques de RV destinés au grand public, mais c'est en 2012 que l'Oculus Rift a véritablement popularisé cette technologie. La RV plonge l'utilisateur dans un monde totalement fictif où il peut interagir avec un environnement simulé grâce à différents dispositifs, tels que des gants, des capteurs ou la reconnaissance vocale (Arnaldi, Fuchs, Guitton, 2023).
La réalité augmentée (RA) repose sur un principe différent : elle ajoute des éléments virtuels au monde réel. Ainsi, l'utilisateur voit le monde à travers un appareil, comme un casque ou des lunettes, auquel se superposent des objets ou des informations virtuelles. Ce principe évoque celui des hologrammes. Certains films d'anticipation, comme *Star Wars* et *Terminator*, avaient déjà exploré cette idée. En intégrant l'interaction avec l'usager, nous obtenons la réalité mixte (RM) que Microsoft a développée avec son casque HoloLens (2016 et HoloLens 2 fin 2019) (VRX, 2023).

Ces technologies impactent les pratiques du géographe et de l'aménageur.

1.2 La dualité des usages des hologrammes en géographie

En pratique, deux utilisations de la réalité mixte se distinguent :

- Le maquettage emploi les hologrammes en amont des projets. Il remplace les maquettes physiques et permet aux usagers équipés de casques de réaliser des simulations en temps réel depuis les bureaux d'études.

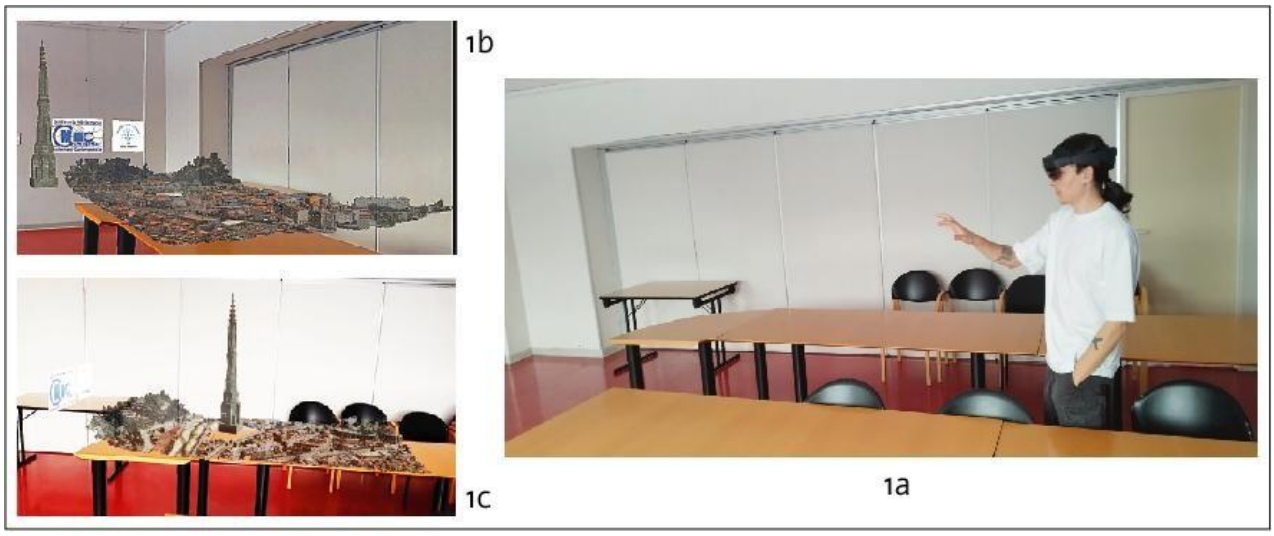

Image 1 : maquette

L'image 1a présente l'utilisateur du casque. Devant lui il n'y a rien. L'image 1b montre ce qu'il voit, en l'occurrence deux objets : une maquette urbaine et une tour. Sur l'image 1c on observe que la tour a été déplacée par l'utilisateur.

- Le fonctionnement du casque en extérieur facilite la visualisation des projets in situ. Cela représente une véritable révolution pour les professionnels, qui peuvent ainsi appréhender l'impact immédiat de leurs projets et intervenir directement sur le terrain.

Image 2 : exemple in situ

L'image 2 illustre ce que perçoit l'utilisateur, c'est-à-dire l'aménagement conçu à sa place exacte dans l'environnement prévu. Les gabions situés sur la route du Boréon, dans les Alpes-Maritimes, nécessitent un embellissement (2a). Les images 2b et 2c montrent l'interface et la main de l'utilisateur en train de sélectionner une plante sur la fenêtre virtuelle de choix. L'image 2d représente l'aménagement final avec la végétation positionnée. Cet exemple réalisé in situ, démontre ainsi ce second aspect de l'utilisation de la réalité mixte.

1.3 Limitations du casque en termes de géoréférencement

Parmi tous les dispositifs de RM disponibles, c'est celui de Microsoft qui est le plus abouti. Il bénéficie de la puissance de développement de la société de Redmond. Actuellement, aucune solution de RM n'intègre de GNSS.

Le casque HoloLens 2 est principalement conçu pour une utilisation en intérieur. Par conséquent, l'absence de géoréférencement n'est pas un problème dans le cadre du maquettage, mais devient un frein dans les applications extérieures où le placement automatique d'objets sur des emplacements prévus serait idéal. Il existe quelques solutions matérielles alternatives, évoquées par de rares articles (Guarese, Maciel, 2019). Elles utilisent des récepteurs GNSS externes communiquant via le Bluetooth. Cependant, leur mise en œuvre peut s'avérer complexe, onéreuse et nécessite des compétences avancées en programmation. Nous examinerons les limitations du Bluetooth dans la deuxième partie.

Une autre possibilité consiste à trianguler la position du casque à partir des réseaux Wi-Fi, une solution moins précise, mais avantageuse là où le signal GNSS est absent, notamment en intérieur (Braga, 2022).

Malgré l'existence de ces quelques prototypes, le géoréférencement demeure un défi pour les applications envisagées avec HoloLens 2. Pour tenter de pallier cette lacune, nous proposons une solution robuste, gratuite et simple que la partie suivante détaille dans sa mise en œuvre technique.

2 Techniques d'intégration du géoréférencement

L'idée principale est d'exploiter les capacités de géolocalisation offertes par les smartphones, tous équipés d'un ou de plusieurs systèmes GNSS (GPS, Glonass, Beidou, Galiléo … souvent combinés afin d'améliorer la précision de la localisation (Zangenehnejad, Gao, 2021)). L'utilisateur positionnera son smartphone devant lui. Les coordonnées récupérées correspondront donc à la position du casque, situé à quelques centimètres de distance. L'azimut indiqué est aligné avec la direction du smartphone, donc face à l'utilisateur.

2.1 Première méthode

L'approche initiale consistait à développer une application Android pour extraire les coordonnées du smartphone et les transmettre au casque via Bluetooth. Android est le système d'exploitation le plus utilisé et le Bluetooth une norme universelle. De plus, des outils de développement, tels que MIT App Inventor 2, sont simples à mettre en œuvre. Cependant, une contrainte majeure liée au Bluetooth qui limite le nombre de connexions simultanées restreignait les possibilités d'expériences collaboratives avec plusieurs casques synchronisés. Face à cette borne, nous avons finalement opté pour une architecture centralisée dont voici le détail.

2.2 Choix technologiques

Pour distribuer les coordonnées à plusieurs casques, nous avons conçu une solution basée sur un code HTML et un serveur. Un dispositif Android se connecte à une page dédiée à partir d'un navigateur internet. La programmation de cette page récupère les coordonnées GNNS de l'appareil puis les sauvegarde sur un serveur. Il suffit alors que le casque se raccorde en wifi au serveur pour avoir accès aux coordonnées. La figure 1 illustre le mécanisme.

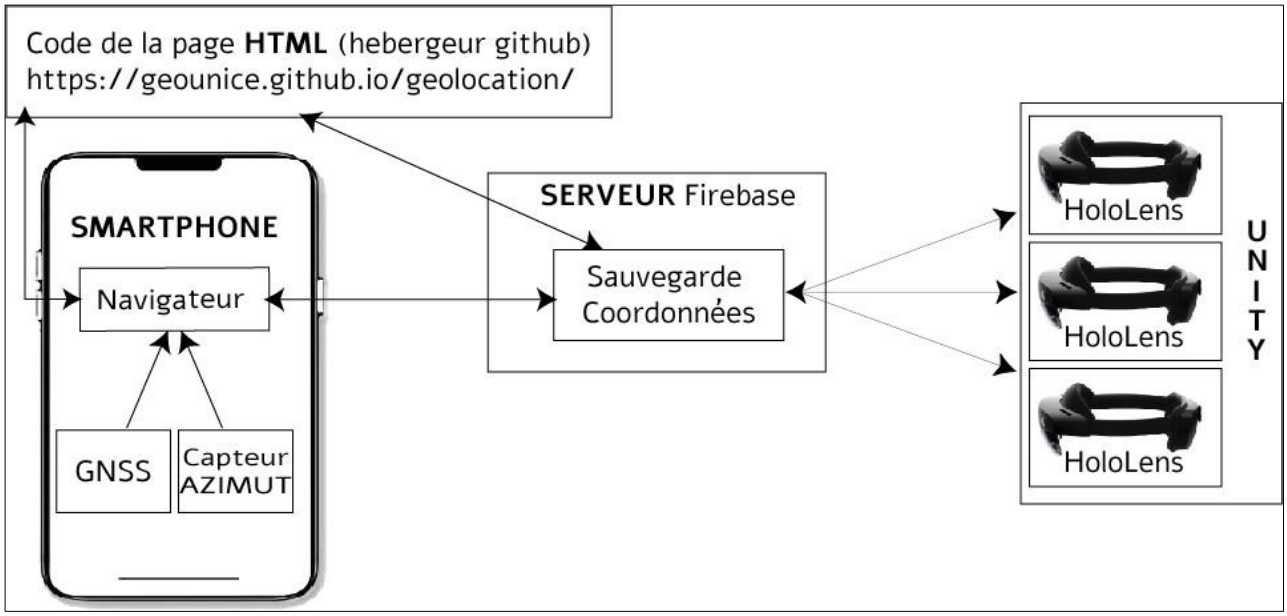

Figure 1 : schéma général

Le code gère la demande de permissions d'accès aux données d'orientation du smartphone, leur affichage et leur sauvegarde sur le serveur. La figure 2 présente l'interface sur le navigateur.

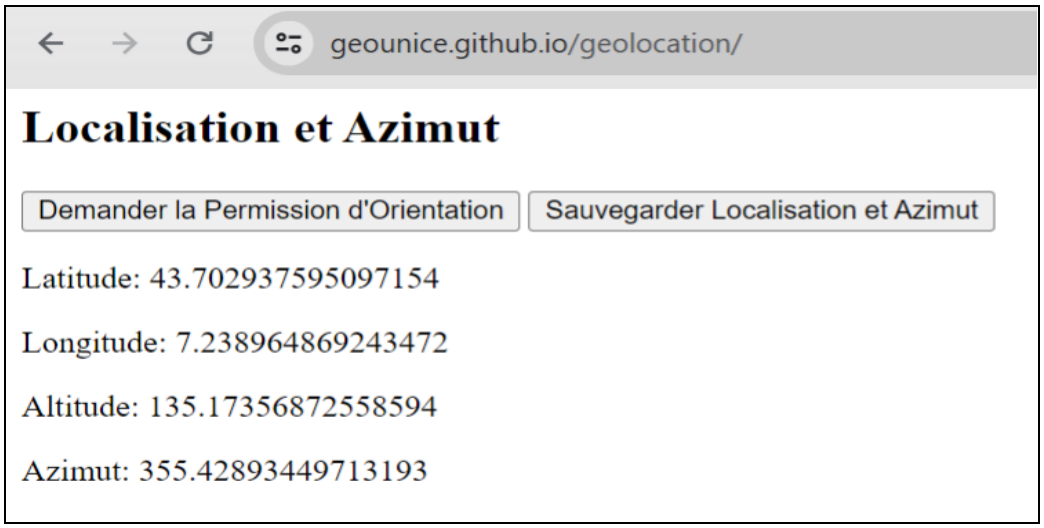

Figure 2 : interface sur l'écran du smartphone à l'adresse : https://geounice.github.io/geolocation/

La solution choisie pour l'hébergement du site est la plateforme GitHub. Ce service open source gratuit est très prisé par les développeurs. Pour le serveur, nous avons opté pour l'outil gratuit proposé par Google : Firebase accessible à l'adresse : https://firebase.google.com/.
Ces choix reposent sur l'universalité et la gratuité de GitHub et Firebase qui offrent des instruments efficaces et accessibles pour le développement et l'hébergement d'applications.
La mise en œuvre du serveur Firebase, standard et bien documentée, ne nécessite pas d'explications particulières. La récupération des données sur le serveur et leur utilisation se font ensuite dans le cadre de la programmation de l'application du casque (réalisée avec le moteur de développement Unity). La figure 3 montre l'interface créée sur Unity pour accéder aux coordonnées.

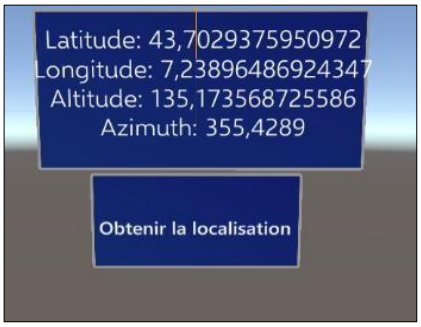

Figure 3 : interface sur la scène Unity

2.3 Mise en œuvre

Le projet est structuré en trois sections applicatives que nous allons aborder en détail.

2.3.1 Le serveur

Comme nous l'avons mentionné, c'est le support Firebase de Google que nous avons choisi. Son utilisation est simple et ne nécessite pas plus de détails dans le cadre de cet article.

2.3.2 L'application Web

La programmation du site se décompose en deux parties principales.
La première repose sur une Application Programming Interface (API) Web standard permettant d'obtenir la position géographique du périphérique qui l'appelle. Cette API, incluse par défaut dans l'environnement JavaScript des navigateurs internet, n'a pas besoin d'être explicitement déclarée. L'API **Geolocation** utilise la méthode **'getCurrentPosition'**. Implémentée dans notre fonction **'window.requestPermission'**, elle retourne un objet **'position'** contenant les informations de géolocalisation (latitude, longitude et altitude).
Pour l'azimut, nous utilisons l'événement **'deviceorientation'**, qui appartient aux API des capteurs d'orientation du navigateur, car cette fonctionnalité n'est pas proposée par l'API **Geolocation**. Notre fonction **'handleDeviceOrientation'** obtient et affiche en temps réel l'azimut.
La deuxième partie concerne l'envoi des coordonnées et de l'azimut récupérés à un serveur. Pour cela, nous employons Firebase Realtime Database, un service de base de données offert par Google Firebase. Ce service gratuit, hébergé dans le cloud, permet de synchroniser les données en temps réel entre les clients et la base de données. Nous configurons et initialisons Firebase dans le header du code HTML. L'API **Firebase** propose la méthode **'set'** (utilisée dans notre fonction **'saveLocationAndAzimuth'**) pour envoyer les données au serveur. Ensuite, nous utilisons la méthode **'get'** pour obtenir les données et les montrer dans le navigateur via notre fonction **'retrieveLocationAndAzimuth'**. Bien que nous aurions pu afficher directement les coordonnées, les récupérer du serveur permet de s'assurer que les données sont à jour et identiques pour tous les utilisateurs accédant à l'application.
Le reste du code est un ensemble de routines classiques apportant les affichages, autorisations et autres fonctionnalités secondaires.
Ce code est hébergé comme dit précédemment à l'adresse https://geounice.github.io/geolocation/.

2.3.3 Développement HoloLens 2

Le programme intégré dans le casque est développé sous Unity. Il ne pose pas de difficulté particulière. Il est écrit en C# et utilise donc la programmation orientée objet.
Tout d'abord, nous structurons nos données en employant une classe nommée **'LocationData'**, qui est composée de quatre variables : latitude, longitude, altitude et azimut, toutes de type double afin de garantir une grande précision. Comme leurs noms l'indiquent, ces champs sont destinés à stocker nos informations géographiques.
Ensuite, une autre classe gère la récupération des données de localisation inscrites sur Firebase. L'appel est fait lorsque l'on clique sur le bouton de la scène Unity par le biais de **'OnClickRetrieveData'**.
C'est **'GetDataCoroutine'** qui va chercher sur le serveur les données. Utiliser une coroutine en C# permet de travailler en asynchrone est donc de ne pas bloquer le déroulement de programme principal pendant qu'elle

s’exécute. On l'emploie souvent pour les requêtes réseau qui dépendent des conditions de communication et de la latence du serveur et peuvent ralentir le fonctionnement de l'application. La commande **UnityWebRequest** effectue une requête **Get** vers l’URL (l’adresse de notre base de données) afin de récupérer les données de localisation.
La Firebase Realtime Database fait appel à un format de données nommé JSON (JavaScript Object Notation) pour le stockage et l'échange d'informations. Unity ne manipulant pas nativement les objets JSON, nous utiliserons une bibliothèque appelée **‘JsonUtility’** pour convertir et intégrer nos données dans notre classe **‘LocationData ’**. Cette transformation permet d’opérer les données au sein d'Unity à travers des objets C#, ce qui facilite l'accès et la gestion des valeurs. Le code source est fourni en annexe.

3 Test et évaluation de la solution

Cette dernière partie présente les résultats de notre recherche.

3.1 Protocole de test

Une fois démarrée, notre application de test, développée sur Unity, établit une connexion avec le serveur, récupère les coordonnées et l’azimut puis les transcrit dans une interface utilisateur. Cette étape nous permet de vérifier la précision du transfert des coordonnées.
Nous comparons l’affichage des coordonnées récupérées dans la fenêtre de notre scène Unity avec les coordonnées GNSS reportées par notre smartphone.

3.2 Résultats obtenus

Nous avons vérifié à plusieurs reprises le bon fonctionnement de notre application. Voici un exemple détaillé d’une de nos mises en œuvre.

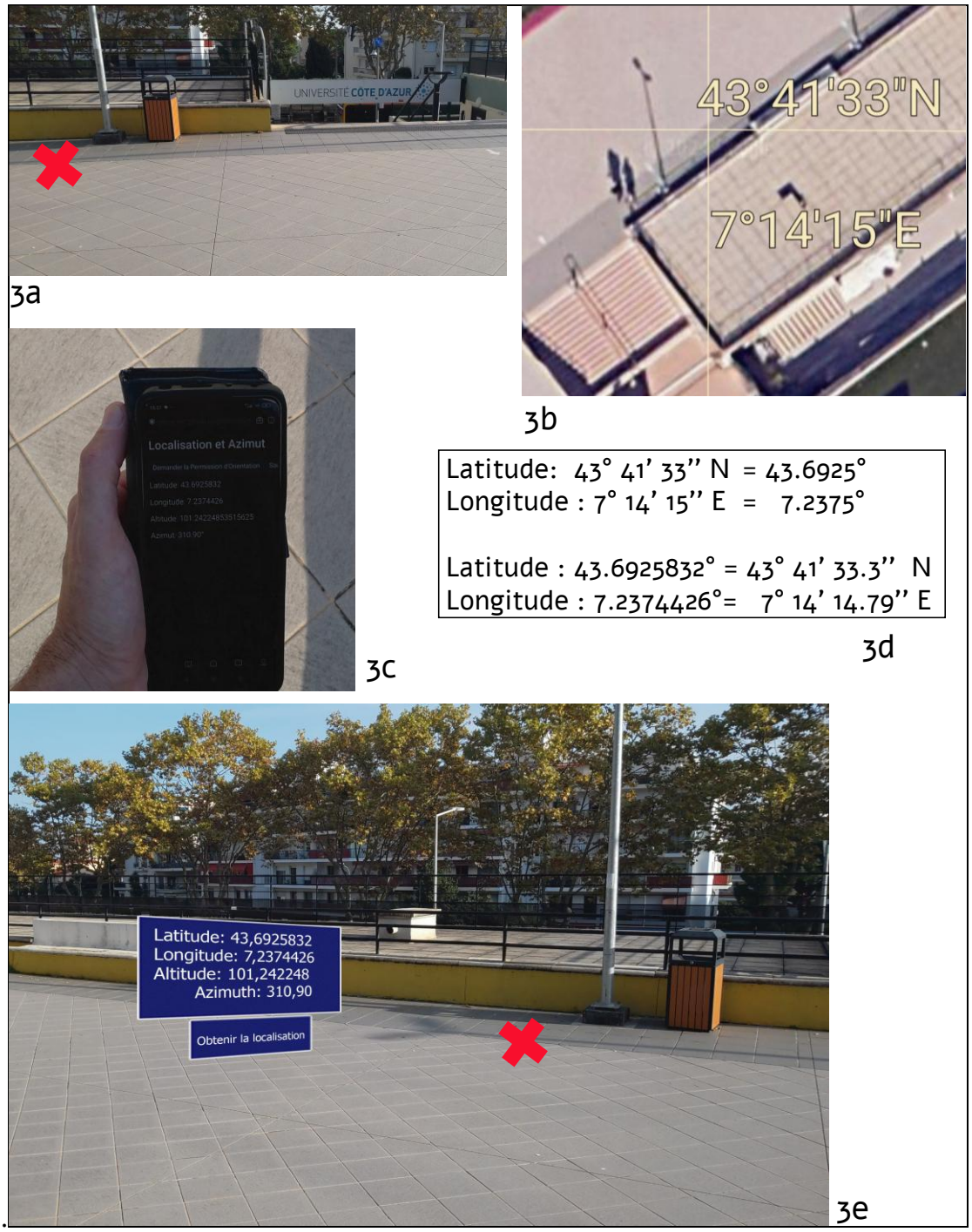

Image 3 : test de la solution

L'image 3a présente le lieu où nous avons récupéré les coordonnées, au niveau de la croix rouge, en l'occurrence le parvis de l'Université Nice Côte d'Azur, campus Carlone.
L'image 3b est issue d'une application Android qui nous fournit la photographie aérienne du lieu et une grille latitude/longitude en format DMS (degré, minute, seconde), tandis que l'image 3c expose les coordonnées que nous récupérons de notre smartphone grâce à notre site. Ces dernières sont en degrés décimaux et l'image 3d propose les conversions. Ainsi nous constatons que la précision de notre récepteur est correcte (avec une marge de quelques mètres, mais cet article n'a pas l'ambition de traiter de la précision du positionnement GNSS).
Enfin, l'image 3e est prise par l'HoloLens 2 après la récupération des coordonnées. Notons que nous nous sommes déplacés à quelques mètres pour l'obtenir. Elle confirme la transmission des coordonnées au casque.

Conclusion et perspectives futures

Suite aux différents tests menés, la solution proposée est pertinente et efficace. Elle permet d'extraire les coordonnées GNSS de n'importe quel smartphone, de les exporter dans une base de données externe et de les récupérer sur un ou plusieurs casques HoloLens2.
La précision dépend de l'acuité du périphérique employé. La fiabilité du système est satisfaisante pour des utilisations en extérieur.
Ce développement est une expérimentation et ne constitue en aucun cas une application finalisée.
En l'absence de signal GNSS, d'autres solutions existent, telles que l'intégration d'Azure Spatial Anchors ou des outils moins professionnels (Tianyu, 2021).
Le futur de cette application permettra le placement automatique d'objets géoréférencés dans le casque. Pour cela, il reste plusieurs étapes à franchir, notamment l'orientation des objets.
Une possibilité supplémentaire serait de coupler le casque à un système d'information géographique (SIG) afin de positionner automatiquement les objets.
L'intérêt croissant pour les applications de réalité mixte nécessitant un géoréférencement précis suggère que Microsoft pourrait intégrer un GNSS dans ses futurs casques HoloLens, bien que cela n'ait pas été confirmé. En attendant, notre solution reste actuellement la seule alternative gratuite disponible.

Bibliographie dans le texte :

Arnaldi, B., Fuchs, P., & Guitton, P. (2023). Introduction à la réalité virtuelle. Repéré à : https://hal.science/hal-00353631/file/vr_intro.pdf.

Braga, G. (2022). HoloLens 2 Unity tutorial (Windows). Repéré à : https://github.com/Gustavobb/HoloLens2-Unity-Tutorial#gps

Brunet, R., Ferras, R., & Théry, H. (1992). *Les mots de la géographie : dictionnaire critique*. Montpellier : Reclus, 1992.

Dupont, J., et al. (2020). L'utilisation des hologrammes pour la visualisation des données géospatiales dans l'aménagement urbain. *Revue Européenne de Géographie*, 15(2), 81-93.

Géoconfluences, Glossaire (n.d.). Géoréférencement. Repéré à : https://geoconfluences.ens-lyon.fr/glossaire/georeferencement

Guarese, R., Maciel, A. (2019). Development and usability analysis of a mixed reality gps navigation application for the microsoft HoloLens. *Computer Graphics International Conference*, Springer, 431-437.

Le Monde, (2019). La géovisualisation, késako?. *Binaire*. Repéré à : https://www.lemonde.fr/blog/binaire/2019/11/18/la-geovisualisation-kezako/

Martin, A., et al. (2019). Applications des HoloLens dans la planification urbaine interactive : Études de cas européennes. *Journal of Urban Technology*, 26(4), 210-225.

Mericskay, B. (2022). *Communication cartographique : Sémiologie graphique, sémiotique et géovisualisation*. ISTE éditions, 2022.

Microsoft Learn, Repéré à : https://learn.microsoft.com/en-us/dynamics365/mixed-reality/guides/pc-app-anchorimprove-hologram-precision et https://learn.microsoft.com/en-us/windows/mixed-reality/design/spatial-mapping

Rzeszewski, M., & Orylski, M. (2021). Usability of WebXR Visualizations in Urban Planning. *International Journal of Geo-Information*, 10(11), 721. Repéré à : https://doi.org/10.3390/ijgi10110721

Tianyu, W. (2021). HoloNav : A Mixed Reality Indoor Navigation System. Repéré à : https://github.com/Tianyu-Wu/HoloNav

VRX (2023). L'histoire de la réalité augmentée et son avenir. Repéré à : https://vrx.vr-expert.com/fr/lhistoire-de-ar-et-son-avenir/

Wijesooriya, P., Farjad, S. M., Stergiou, N., & Mastorakis, S. (2023). Investigating the Characteristics and Performance of Augmented Reality Applications on Head-Mounted Displays: A Study of the HoloLens Application Store. *IEEE International Conference on Communications Workshops (ICC Workshops)*. Repéré à : https://par.nsf.gov/biblio/10406321

Zangenehnejad, F., Gao, Y. (2021). GNSS smartphones positioning : advances, challenges, opportunities, and future perspectives, *SpringerOpen*, *Satell Navig* 2, 24, 2021. Repéré à : https://doi.org/10.1186/s43020-021-00054-y

ANNEXE :

Code du script Unity de récupération des données

```
using System;
using System.Collections;
using UnityEngine;
using UnityEngine.Networking;

[Serializable]
public class LocationData
{
    public double latitude;
    public double longitude;
    public double altitude;
    public float azimuth;
    public decimal newMetricLatitude;
    public decimal newMetricLongitude;
    public decimal newMetricAltitude;
}

public class LocationDataPublisher : MonoBehaviour
{
    public static event Action<LocationData> OnDataRetrieved;
    public static void PublishData(LocationData data)
    {OnDataRetrieved?.Invoke(data);}
}

public class recup_donnees : MonoBehaviour
{
    public void OnClickRetrieveData()
    {StartCoroutine(GetDataCoroutine());}

    IEnumerator GetDataCoroutine()
    {
        string databaseURL = "https://geolocalisation-unice-default-
rtdb.europe-west1.firebasedatabase.app/locations/lastLocation.json";
        UnityWebRequest www = UnityWebRequest.Get(databaseURL);
        yield return www.SendWebRequest();

        if (www.isNetworkError || www.isHttpError)
        {
            Debug.LogError("Erreur de récupération : " + www.error);
        }
        else
        {
            LocationData locationData =
JsonUtility.FromJson<LocationData>(www.downloadHandler.text);
            Debug.Log("Latitude: " + locationData.latitude);
            Debug.Log("Longitude: " + locationData.longitude);
            Debug.Log("Altitude: " + locationData.altitude);
            Debug.Log("Azimuth: " + locationData.azimuth);

            if (DataManager.Instance != null)
            {
                DataManager.Instance.locationData = locationData;
            }

            LocationDataPublisher.PublishData(locationData);
        }
    }
}
```