UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS FACULTAD DE INGENIERÍA

CARRERA DE CIENCIAS DE LA COMPUTACIÓN

PRACTICA CALIFICADA Nro 1

PROGRAMACION CONCURRENTE Y DISTRIBUIDA

ALUMNO: Diaz Huarcaya Henry Josue. **CÓDIGO: U20201C579**

PROFESOR: Carlos Alberto Jara García

Lima, 2024-2

TITULO	**Handling Concurrency in Behavior Trees**	**VanityX: An Agile 3D Rendering Platform Supporting Mixed Reality**	**Deep learning(s) in gaming disorder through the user-avatar bond: A longitudinal study using machine learning**
AUTORES	Michele Colledanchise, Lorenzo Natale	Ivan Zoraja, Mirjana Bonkovic, Vladan Papic, Vaidy Sunderam	Vasileios Stavropoulos, Daniel Zarate, Maria Prokofieva, Noirin Van de Berg, Leila Karimi, Angela Gorman Alesi, Michaella Richards, Soula Bennett, Mark D. Griffiths
AÑO	2021	2023	2023
MOTIVACIÓN	La necesidad de manejar comportamientos concurrentes en robots y agentes autónomos, mejorando la modularidad y la reutilización de los árboles de comportamiento (BTs) en aplicaciones de robótica y videojuegos.	La motivación detrás del desarrollo de VanityX fue la dificultad y el tiempo que requiere la integración de componentes comerciales y de código abierto en soluciones de realidad extendida (XR). Además, se identificó la necesidad de cumplir con requisitos específicos de los usuarios, como el renderizado volumétrico, que no estaban disponibles en los activos estándar de Unity.	La creciente preocupación por el trastorno por videojuegos (GD) y la necesidad de identificar factores de riesgo a través de la relación entre los jugadores y sus avatares en el entorno de los videojuegos.
PROBLEMA	Los problemas de concurrencia en BTs, como condiciones de carrera y bloqueos, que limitan la efectividad de la composición paralela de comportamientos.	La integración de múltiples componentes en sistemas XR puede resultar en una calidad cuestionable, ya que el sistema completo puede estar limitado por el componente más débil. Esto, junto con la falta de funcionalidades	La dificultad para predecir el riesgo de GD en jugadores basándose en su relación con los avatares, lo que puede llevar a diagnósticos tardíos o inexactos.

		específicas en soluciones existentes, llevó a la necesidad de crear una plataforma propia.	
PROPUESTA DE SOLUCIÓN	Introducción de nuevos nodos de BT que permiten la sincronización de recursos y el manejo de la concurrencia, mejorando la previsibilidad y el rendimiento de los comportamientos.	Se propuso desarrollar VanityX, una plataforma de renderizado 3D y computación en tiempo real basada en DirectX 12, que permite la colaboración eficiente de equipos médicos y soporta tanto el renderizado basado en superficies como el renderizado volumétrico.	Utilizar algoritmos de aprendizaje automático (ML) para analizar datos longitudinales sobre la identificación del usuario con su avatar, la inmersión y otros factores, con el fin de predecir el riesgo de GD.
ALGORITMOS USADOS, PSEUDOCÓDIGO	Se presentan nuevos nodos de control que implementan técnicas de sincronización y gestión de recursos. El pseudocódigo específico no se detalla en el resumen, pero se puede encontrar en el documento completo.	El paper no proporciona pseudocódigo específico, pero menciona el uso de algoritmos efectivos para la gestión de recursos gráficos y el procesamiento paralelo explícito, lo que sugiere un enfoque en la optimización del rendimiento.	Se utilizaron algoritmos de ML como Random Forests, Logistic Regression y Support Vector Machines (SVM).
DESCRIPCIÓN DE SERVICIOS, BIBLIOTECAS USADAS, CÓDIGO, LENGUAJE DE PROGRAMACIÓN USADOS PARA IMPLEME	Se proporciona una implementación de código abierto de la nueva formulación de BTs. El lenguaje de programación utilizado no se especifica, pero se asume que es compatible con entornos de robótica como ROS.	Lenguajes de programación: C++20 y HLSL 6. Bibliotecas y servicios: La arquitectura de VanityX incluye un motor 3D, servicios de computación y renderizado, y una API basada en DirectX 12. Se menciona el uso de OpenXR para la interoperabilidad con dispositivos de realidad mixta.	Lenguaje de Programación: R. Bibliotecas: Tidymodels para la implementación de modelos de ML, DMwR para la técnica SMOTE (Synthetic Minority Over-sampling Technique) para balancear los datos. Servicios: Se utilizó una aplicación de monitoreo móvil llamada Aware Light para recopilar datos sobre el uso del dispositivo.

NTAR EL MODELO			
OPINIÓN CRÍTICA	La programación concurrente en BTs puede mejorar la eficiencia y la capacidad multitarea, pero también introduce complejidades que pueden dificultar la depuración y el mantenimiento del código. Es crucial un diseño cuidadoso para evitar conflictos.	La programación concurrente ofrece varios beneficios, como la mejora del rendimiento y la eficiencia en el uso de recursos, especialmente en aplicaciones que requieren procesamiento en tiempo real, como el renderizado 3D. Sin embargo, también puede introducir complejidades adicionales, como la gestión de la sincronización y la posibilidad de errores difíciles de depurar. En el contexto de VanityX, la programación concurrente parece ser esencial para alcanzar los objetivos de rendimiento y escalabilidad, pero requiere un diseño cuidadoso para evitar problemas de concurrencia.	La programación concurrente puede ofrecer beneficios significativos en términos de eficiencia y velocidad en el procesamiento de datos, especialmente en el análisis de grandes volúmenes de información. Sin embargo, también puede introducir complejidades adicionales, como la gestión de recursos y la sincronización, lo que puede llevar a errores difíciles de depurar. Es crucial que los desarrolladores tengan un buen entendimiento de la programación concurrente para maximizar sus beneficios y minimizar sus desventajas.
LINK DE ACCESO	https://doi.org/10.1109/TRO.2021.3125863	https://doi.org/10.3390/app13095468	https://doi.org/10.1556/2006.2023.00062