Multi-View Environment, un software de fotogrametría sobre contenedores *Docker*

Raymundo Dominguez-Colin, Lil Maria Rodriguez-Henriquez, María HortensiaAlmaguer-Cantú, Juan Roberto Hernández-Garibaí

Universidad Juárez Autónoma de Tabasco. División Académica de Ciencias Básicas. Cunduacán, Tabasco, México.

Resumen

Multi-View Environment (*MVE*) es una aplicación *software* que realiza reconstrucción geométrica de imágenes digitales. Está diseñado para crear modelos en tres dimensiones (3D). *MVE* es un programa diseñado para funcionar en los sistemas operativos más conocidos: *Linux*, *MacOS* y *Windows.* Sin embargo,el procedimiento para su instalación en este último es demasiado complejo y no cuenta con soporte técnico de los programadores, limitando el potencial del *software*para usuarios de *Windows.* Los contenedores *Docker* son una alternativa eficiente para el funcionamiento de *MVE* sobre este sistema operativo. En este artículo se presenta un método para la instalación, compilación y ejecución de *MVE* sobre contenedores *Docker* en *Windows*. El resultado es un ciclo completo de ejecución eficiente de *MVE* sin tener que recurrir a la emulación de una máquina virtual convencional.

Abstract

Multi-View Environment (MVE) is a software application that performs geometric reconstruction of digital images. It is designed to create three-dimensional models (3D). MVE is a program designed to run on the most popular operating systems: Linux, MacOS and Windows. However, the procedure for its installation on the latteris too complex and does not have technical support from the programmers, limitingthe potential of the software for Windows users. Docker containers are an efficient alternative for running MVE on this operating system. This article presents a methodfor installing, compiling, and running MVE on Docker containers on Windows. The result is a complete cycle of efficient MVE execution without having to resort to conventional virtual machine emulation.

Palabras clave - Fotogrametría, Multi-View Environment, Docker, MVS-Texturing.
Keywords – Photogrammetry, Multi-View Environment, Docker, MVS-Texturing.

1. INTRODUCCIÓN

El procesamiento de imágenes para generar modelos tridimensionales es un área de la computación que se ha popularizado en la última década. El acceso a dispositivos con capacidad de tomar fotografías digitales de buena calidad, sumadoal aumento en el procesamiento de los equipos de cómputo actuales, portátiles o de escritorio, han contribuido al desarrollo de aplicaciones útiles en la generación de modelos 3D. La fotogrametría es una rama de la visión por computadora que se basa en el procesamiento de imágenes digitales. Dichas imágenes corresponden aobjetos físicos, tomadas desde distintos ángulos y con un cierto nivel de traslape. Una de las ventajas de esta área es que no es necesaria una cámara profesional, ya que basta con la cámara de un equipo celular y un equipo de cómputo.

El modelado tridimensional consiste en cuatro etapas principales: la estructura a partir del movimiento (*Surface from Motion*) para generar una nube de puntos dispersa, el procesamiento de multi-vistas (*Multi-view processing*), la creación de superficies (*surface reconstruction*) y finalmente la generación de texturas (*texture*). Existe una gran variedad de herramientas disponibles que permiten el procesamiento de imágenes para la generación de modelos 3D. Muchas de éstas están enfocadas en imágenes aéreas y otras en imágenes terrestres. En *Internet* puede encontrarse una gran variedad de paquetes de *software* disponibles para el procesamiento de imágenes terrestres. Muchos de éstos son productos comerciales, como *Photomodeler* [1],

IMAGINE Photogrammetry [2], *Bentley Context Capture* [3], *Agisoft* [4], *3DF Zephyr* [5], *Capturing Reality* [6], entre muchosotros. También hay productos que son gratuitos como *COLMAP* [7] [8], *Meshroom* [9] y existen incluso opciones de código abierto como *MicMac* [10] o *Multi-View Environment* (MVE) [11].

Los requerimientos tecnológicos de los productos mencionados varían de acuerdo con sus desarrolladores. Algunos como *COLMAP*, *Meshroom* requieren de tarjetasgráficas especializadas como NVIDIA y CUDA para funcionar correctamente. Lo anterior representa un inconveniente si no se cuenta con equipos que cubran dichos requerimientos. No obstante, existen alternativas igual de eficientes que no requieren de recursos de cómputo especializados. Una de éstas es *Multi-View Environment*, la cual es gratuita y de código abierto. Esta herramienta se encuentradisponible para sistemas *Linux*, *MacOS* y *Windows*. Sin embargo, el proceso para hacerlo funcionar en este último es complejo e incluso no tiene el soporte técnico de parte de los desarrolladores [12]. La instalación de *MVE* sobre *Linux* y *MacOS* es directa y se puede realizar a través del administrador de paquetes *Snap*. Desafortunadamente, los usuarios que no dominen o manejen *Linux* no podrán utilizar *MVE* directamente. Algunas opciones para su manejo sería la de instalar unamáquina virtual que les permita emular *Linux* o bien, instalarlo y contar con un arranque dual en su equipo. Ambas opciones requieren de una gran experiencia y conocimientosen la instalación y configuración de *software*. Además, virtualizar un sistema operativo completo (SO), si bien es funcional, trae consigo algunas desventajas como la de tener que compartir recursos con el SO invitado, tales como el número de procesadores, memoria *RAM*, etc. Aunado a lo anterior, los procesos fotogramétricos suelen requerir demasiados recursos del CPU y de memoria.Debido a lo cual, el procesamiento de imágenes en un entorno virtual podría llegar a ser demasiado costoso para un equipo que no cuente con suficientesrecursos de cómputo.

Una solución eficiente es ejecutar *MVE* sobre un contenedor *Docker*. Éstos se hanposicionado como una alternativa a la emulación tradicional de sistemas operativoscompletos, ya que permiten emular el *kernel* básico de cualquier SO. Lo anterior facilitala ejecución e incluso el desarrollo de aplicaciones que funcionan en sistemas operativos distintos al anfitrión. No obstante, hasta donde se conoce, no hay un procedimiento o serie de pasos a seguir para instalar *MVE* sobre *Docker*. En este artículo se presenta un procedimiento para realizar una instalación y ejecución exitosa de *Multi-View Environment* en un contenedor *Docker* sobre *Windows*. Con el procedimiento mostrado, se podrá completar un flujo de procesamiento completo de *MVE*. Este documento está dirigido a usuarios de *Windows* que no cuenten con mucha experiencia en el manejo de *Linux* y que además estén interesados en el uso de *MVE* para el procesamiento de imágenes ygeneración de modelos 3D.

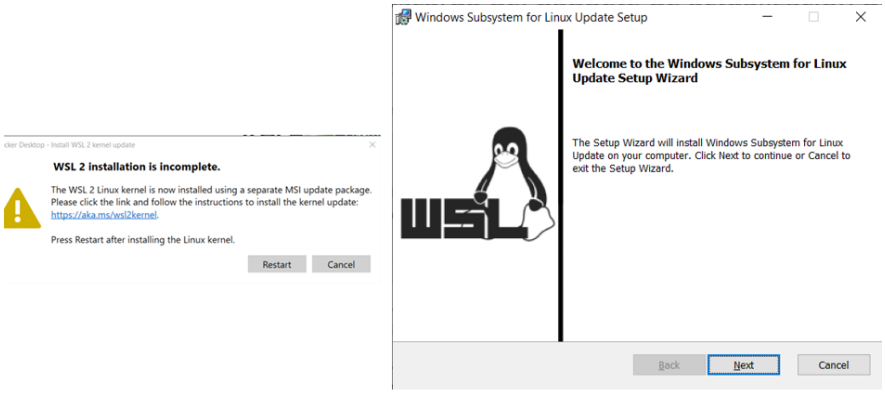
2. MÉTODO DE INSTALACIÓN

En esta sección se describirán los pasos que deben de seguirse para instalar correctamente *MVE* sobre *Docker* en *Windows*. Primero se indicará cómo instalar *Docker* y posteriormente se mostrará cómo descargar, instalar y ejecutar *Multi-ViewEnvironment* sobre el entorno virtual creado. Finalmente, se instalará un tercer programa que se llama *TexRecon* o también MVS-Texturing [13]. Este último se encarga de la generación de la textura de los modelos 3D. Esta sección da un panorama general de *Docker*, su instalación y su manejo básico. Sin embargo, no es un manual de usuario de *Docker*. Para una mejor y más completa referencia de esta plataforma, el lector deberá referirse a las páginas oficiales [14].

ISSN: 2007-4786

a. Docker

Docker es una plataforma de desarrollo, distribución y ejecución de aplicaciones. Permite trabajar en alguna aplicación de manera local, aún si ésta es nativa de otrosistema operativo. Los pasos para la instalación de esta plataforma de desarrollo pueden encontrarse en el sitio oficial [14]. Se tiene que descargar el instalador y seguir los pasos indicados por el mismo, los cuales son muy directos y sería redundante describirlos en este documento. La única recomendación será la de nodejar de instalar el Subsistema de *Windows* para *Linux* (SWL), el cual permitirá ejecutar un entorno de *GNU/Linux* básico y completamente funcional. De hecho, al momento de instalar *Docker* es posible que el sistema muestre una ventana con un mensaje que indique que la instalación está incompleta, ya que falta el *SWL* (Figura1(a)). En el mismo mensaje se encuentra un enlace que dirige al usuario al sitio dedescarga e instalación de dicho programa (Figura 1(b)).



(a) (b)

Figura 1. Instaladores. (a) Al instalar *Docker* se le solicita la instalación del *WSL*. (b) El instaladorde *WSL* es muy directo y le guía a través de una secuencia de ventanas para su instalación.

b. Imágenes de *Docker* y contenedores

Los componentes más importantes de los entornos *Docker* son la *imagen* y los contenedores mismos. De hecho, en *Docker* todo está basado en sus imágenes. Una imagen en *Docker* (no confundir en lo absoluto con el concepto de *imagen* digital) es una plantilla de solo lectura que contiene todo lo necesario para el despliegue yejecución de contenedores [15]. Técnicamente es una colección de capas de archivos que encapsulan todo lo necesario (dependencias, código fuente y bibliotecas) para establecer un entorno de contenedores funcional. Por otro lado, uncontenedor es una elemento ejecutable e independiente que permite la ejecución de aplicaciones. Así que, teniendo una imagen *Docker* de *Linux*, un contenedor sería una estancia de dicha imagen que en la cual se podrá manejar un ambiente de línea de comandos de *Linux*.

c. Crear una imagen de *Linux* en *Docker*

Una imagen de *Linux* en *Docker* permitirá el funcionamiento de uno o más contenedores. Para crear esta imagen, se debe de escribir en la terminal de *Windows* el siguiente comando:

:> docker pull ubuntu

Al ejecutar este comando por primera vez, se descarga la última versión de *Ubuntu*de los repositorios de *Docker* y se instala en una imagen con nombre *ubuntu*. Paralistar las imágenes creadas (Figura 2), se utiliza el comando:

:> docker images

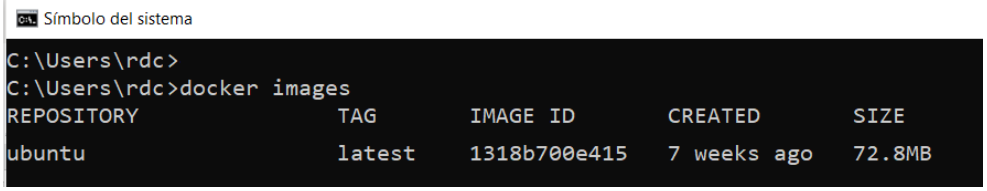


Figura 2. Imágenes de *Docker*. El comando *docker images* muestra las imágenes creadas.

Ahora hay que crear un contenedor de la imagen, que es en donde se van a instalartodas las herramientas necesarias para el funcionamiento de *MVE*.

d. Contenedores *Docker*

Un contenedor de Docker es un paquete que contiene todas las dependencias y utilidades necesarias para la ejecución y el desarrollo de programas. Los contenedores *Docker* se manejan de manera independiente, ejecutando varios procesos y aplicaciones por separado [14]. De esta manera hacen un uso óptimo de la infraestructura del sistema anfitrión, conservando la seguridad que tendría con sistemas separados. Sin embargo, se requiere que el contenedor comparta archivos con el sistema local, lo cual se explicará a continuación.

e. Crear un contenedor *Docker* con un directorio compartido con elanfitrión

La comunicación entre el sistema anfitrión y el contenedor *Docker* es fundamental.Se necesita de un medio en el que sea posible tanto el envío como la recuperaciónde archivos entre las dos partes. Una de las maneras más sencillas es la de crear un directorio compartido accesible para ambos sistemas. Los pasos para hacerlo son los siguientes:

1. Crear un directorio en el equipo anfitrión. En algún punto de la unidad local odisco C, cree un directorio. Éste fungirá como raíz para compartir archivos con elcontenedor. A manera de mostrarlo con un ejemplo, a este directorio se le llamará *folderCompartido* (Figura 3).

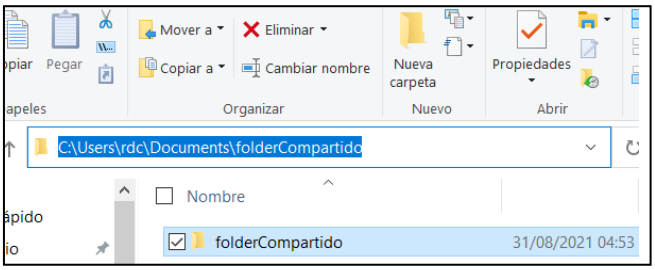


Figura 3. Creación del directorio compartido. (a) En el SO anfitrión se crea un directorio que servirápara comunicarse con el contenedor.

2. Contenedor compartido. Para crear un contenedor compartido, se debe de indicar la ruta y el nombre del directorio anfitrión y un nombre para su símil en elcontenedor. Dando seguimiento al ejemplo, se le llamará también *folderCompartido* a este último, aunque no hay ninguna restricción para que el nombre del directorio local y el de contenedor sean diferentes. El comando de creación del contenedor es el siguiente:

:>docker run -dit -P –name contenedor -v "C:\Users\rdc\Documentos\folderCompartido":/folderCompartido ubuntu

Donde el parámetro -d permitirá ejecutar el contenedor en segundo plano, -tasigna un *pseudo-teletypewriter* (*TTY*) para que se pueda insertar texto dentrode la terminal y -i mantiene el sistema *STDIN* (*Standar Input*) en ejecución; -P hace públicos los puertos de comunicación al anfitrión; --name espera un nombre para el contenedor (que en este ejemplo es "contenedor"); -vse refierea la ruta del volumen del directorio local y ubuntues la imagen que será utilizada.Observe que la ruta del directorio compartido va entre comillas dobles, evitandode esta manera errores en caso de que haya espacios en el nombre de las rutasde los directorios de *Windows* del usuario.

3. Guardar el ID generado. El comando del punto anterior devuelve un ID de 64 caracteres alfanuméricos. De este ID, sólo se requiere conservar los primeros cuatro caracteres. En este ejemplo, el ID generado es:

:>ddf22053e4cf4c80acd1b6298062fa6b02375b536ca3a91acc5fd87e71b794da

Así que se debe de tomar la cadena ddf2.

4. Inicializar el contenedor. Para inicializar por primera vez el contenedor compartido, se escribe el siguiente comando:

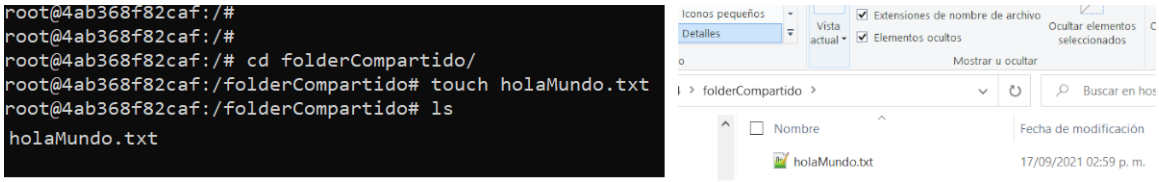
:> docker attach ddf2

Considere que el ID ddf2 será diferente en el equipo del usuario donde realice estos pasos. Finalmente, con este comando se entra al contenedor de *Docker*. Observe que el *promt* de la terminal está como administrador (*root)* por defecto(Figura 4). En este punto se pueden ejecutar algunos comandos de *Linux*, talescomo ls (similar al dir *de Windows*), pwd (para conocer la ruta completa del directorio actual), touch (crea archivos), etc. Para una referencia más completa acerca de comandos de *Linux*, puede dirigirse a tutoriales muy completos que hay en la red [16]. Al consultar el contenido del directorio, compruebe que se hamontado el directorio *folderCompartido*.

Figura 4. Contenedor compartido. Ahora al listar los directorios, debe de aparecer el directorio **folderCompartido** montado en el sistema de *Linux*.

A continuación, debe de verificarse que se puedan crear y modificar archivos enambos espacios compartidos. Haga la prueba de crear un documento de texto en el contenedor y modificarlo en el sistema anfitrión. Acceda al directorio *folderCompartido* y cree el archivo con el comando touch(Figura 5(a)):

root:/# cd folderCompartido
root:/# touch holaMundo.txt



(a) (b)

Figura 5. Comprobación de comunicación. (a) Se accede al directorio del contenedor y se crea elarchivo de texto. (b) El archivo de texto debe de aparecer en el directorio local.

Ahora compruebe que exista el archivo en el directorio del sistema anfitrión (Figura 5(b)). Intente modificarlo con algún editor (Figura 6(a)), guarde los cambios y ahora verifique dicho cambio en el contenedor con el comando cat:

root:/# cat holaMundo.txt

(a) (b)

Figura 6. Comprobando modificación. (a) En *Windows* se modifica el archivo con algúneditor de texto. (b) Dentro del contenedor, el contenido del archivo puede visualizarse con **cat**.

A este punto ya se ha instalado adecuadamente el contenedor con un directorio compartido. Ahora se procederá a la instalación de los programas *Multi-View Environment* y *MVS-Texturing*. Puede salir del contenedor con Ctrl + D o escribiendo el comando exit.

Instalación de *software* en el contenedor

En esta sección se describirán los pasos para la descarga, compilación y ejecuciónde *Multi-View Environment* y *MVS-Texturing*. Debido a que el entorno de *Linux* del contenedor contiene sólo los elementos y programas básicos, es necesario instalarun conjunto de paquetes y bibliotecas para compilar estos dos programas

f. Inicializar el contenedor *Docker*

Los contenedores *Docker* deben de inicializarse antes de poder ejecutarse. La listade los contenedores que existen (Figura 7) puede consultarse con:

:> docker ps -a

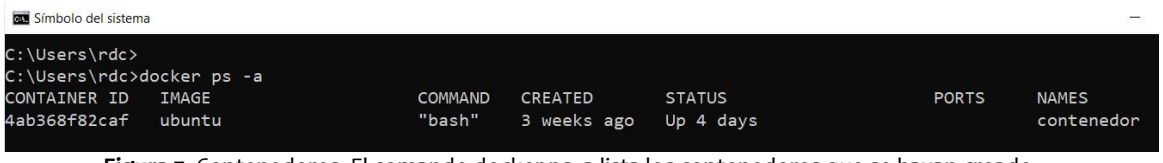


Figura 7. Contenedores. El comando docker ps -a lista los contenedores que se hayan creado.

El comando devuelve diversa información del contenedor: un número identificador (ID), el nombre de la imagen, el o los comandos con los que se iniciará, el tiempo de creación, el estatus y el nombre del contenedor. Para entrar de nuevo al contenedor compartido, el primer paso es iniciarlo y posteriormente ejecutarlo. En una terminal cmd escriba el siguiente comando:

:> docker start contenedor
:> docker exec -it contenedor /bin/bash

Considere que "contenedor" es el nombre que se la ha dado al contenedor mismo en este documento. El usuario puede seleccionar cualquier otro nombre que le parezca más adecuado.

g. Actualizar repositorios e instalación de paquetes necesarios

Para poder comenzar a instalar los paquetes, se deben de actualizar los repositoriosde *Linux*:

root:/# apt-get update

En este paso puede ocurrir que haya problemas con los repositorios y que muestreun mensaje de error en algunas de las direcciones. Lo que procede realizar aquí será reiniciar el equipo y también reiniciar *Docker*.

h. Paquete *build-essential*

El considerado mega-paquete build-essential contiene una lista de enlaces hacia otros paquetes que serán instalados como dependencias. Éste proporciona todo lonecesario para compilar programas desarrollados en C y C++. Sin embargo, comose verá más adelante, se necesitarán algunas bibliotecas adicionales. El comando para instalarlo es:

root:/# apt-get install build-essential

i. Paquete *make*

La utilidad make de *Linux*, dirigida a programadores y desarrolladores, facilita la instalación de herramientas de desarrollo, evitando tener que visitar el sitio web o los repositorios de éstas. Para instalar make se tiene que hacer:

root:/# apt-get -y install make

j. Compiladores de C++

MVE está desarrollado en C++. Aunque el paquete build-essential instala compiladores de C y C++, se tienen que agregar dos compiladores adicionales: g++-10 y gcc-10:

root:/# apt-get install g++-10
root:/# apt-get install gcc-10

k. Paquete *git*

Este paquete permitirá descargar contenido de repositorios o direcciones *ftp*.

root:/# apt-get install git

l. Bibliotecas de imágenes y OpenGL

MVE depende de un mínimo de bibliotecas externas. Necesita la instalación de tresbibliotecas de imágenes: *libjpeg* [17], *libpng* [18] y *libtiff* [19] y, además, de *OpenGL*[20]. Estos paquetes se instalan con:

root:/# apt-get install libjpeg-dev **root:/#** apt-get install libpng-dev
root:/# apt-get install -y libtiff-dev
root:/# apt-get install libglu1-mesa-dev freeglut3-dev mesa-common-dev

m. Descargar y compilar *MVE*

Para descargar MVE lo más recomendable es hacerlo desde su sitio *web*. Primero hay que ir al directorio *usr/local/bin* y desde allí realizar la descarga con *git clone*:

root:/# cd /usr/local/bin
root:/# git clone https://github.com/simonfuhrmann/mve.git

El comando anterior descarga un directorio de nombre mve, el cual contiene los archivos y códigos fuente del programa. Lo siguiente es entrar a este directorio y desde ahí se realizará la compilación. Para hacerlo sin errores, se debe de modificarel comando indicado por los desarrolladores: make -j8. Sin embargo, se tienen que agregar algunos cambios de etiquetas de los compiladores. El comando funcional es:

root:/# cd mve
root:/# make CC=gcc-10 CPP=g++-10 CXX=g++-10 LD=g++-10 -j8

n. Variables de entorno

Aunque se modifiquen, el contenedor *Docker* no almacena las variables de entornoal salir y volver a entrar en él. Así que, para ejecutar MVE se tendrá que agregar como prefijo la ruta /usr/local/bin/mve/apps/, que es en donde se encuentranlos archivos ejecutables del programa.

3. INSTALACIÓN DE *MVS TEXTURING*

MVS-Texturing permitirá la generación de textura a partir de los archivos creados por *MVE* [21]. Se debe de regresar al directorio raíz y posteriormente realizar una actualización de compiladores:

root:/# cd ~/..
root:/# update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10 100 --slave /usr/bin/g++ g++ /usr/bin/g++-10 --slave /usr/bin/gcov gcov /usr/bin/gcov-10

Ahora se descarga el programa MVS con:

root:/# git clone https://github.com/nmoehrle/mvs-texturing.git

y una vez descargado, realizar:

root:/# cd mvs-texturing
root:/# mkdir build && cd build && cmake ..

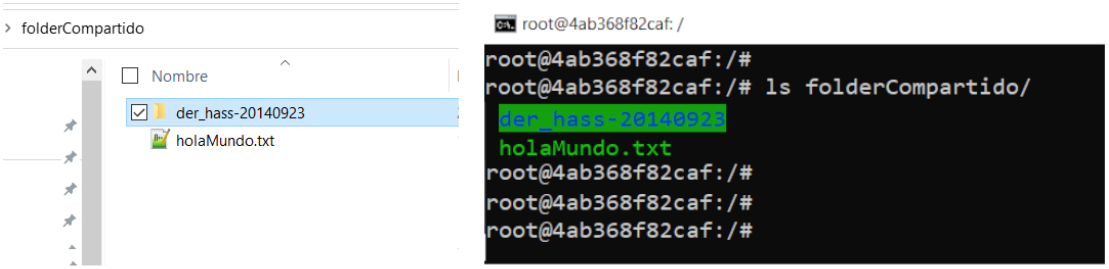
y se compila con:

root:/# make CC=gcc-10 CPP=g++-10 CXX=g++-10 LD=g++-10

De esta manera ya ha quedado instalado el programa que servirá para crear la textura de los modelos tridimensionales. Con estos dos programas se tienen loselementos necesarios para el procesamiento de los modelos tridimensionales. El procedimiento para procesar imágenes digitales y generar modelos tridimensionalesquedan fuera del objetivo de este documento. Sin embargo, el usuario puede consultar las indicaciones en el sitio web de los desarrolladores [11] u otras guías para el manejo de *MVE*.

4. RESULTADOS

El flujo completo de procesamiento de *MVE* y *TexRecon* es muy directo. Se requierede un conjunto de imágenes digitales de algún objeto físico en particular. De no contar con algún conjunto propio, puede descargar algunos que se encuentran disponibles en la sección *Datasets* de [23]. Para este ejemplo, se utilizará el archivoder_hass-20140923. Descargue y ubique las imágenes en el directorio*folderCompartido* del sistema anfitrión (Figura 8(a)).



(a) (b)

Figura 8. Archivo de prueba. (a) Descargue el archivo **der_hass-20140923.tar.gz** y descomprima enel folder compartido (b) El directorio se visualiza dentro del contenedor.

Ahora, en el contenedor cree un nuevo directorio en /home con un nombre cualquiera, sin espacios. Posteriormente se deben de mover las imágenes de prueba del folder compartido al directorio recién creado. Finalmente, cambie de directorio a donde se encuentra el archivo de imágenes. Lo anterior se ejecuta conlos siguientes comandos:

root:/# mkdir home/imgs
root:/# mv folderCompartido/der_hass-20140923/ home/imgs/
root:/# cd home/imgs/der_hass-20140923/

Donde imgs es el nombre que se le ha dado al directorio en este ejemplo. El comando mv hace la función de cortar y pegar de Windows. Una vezen el directorio der_hass_20140923, deberá de ejecutarse la siguiente secuencia de comandos que realizan el procesamiento de MVE:

root:/# /usr/local/bin/mve/apps/makescene -i . scene
root:/# /usr/local/bin/mve/apps/sfmrecon scene **root:/# /usr/local/bin/mve/apps/dmrecon -s2** scene
root:/# /usr/local/bin/mve/apps/scene2pset -F2 scene scene/pset-L2.ply
root:/# /usr/local/bin/mve/apps/fssrecon scene/pset-L2.ply scene/surface-L2.ply
root:/#/usr/local/bin/mve/apps/meshclean-t10 scene/surface-L2.plyscene/surface-L2-clean.ply

El directorio scene es en donde se almacenarán los archivos y directorios de salidade MVE. Al igual que en los casos anteriores, el nombre de este directorio puede cambiar a elección del usuario.

a. Generación de textura

Los comandos arriba citados generan diversos archivos especiales de los modelos3D digitales. Sin salir del directorio actual, el siguiente comando realizará la creaciónde la textura del modelo:

root:/#/mvs-texturing/build/apps/texrecon/texrecon scene**::undistorted**scene/surface-L2-clean.ply textured

Todos los archivos de salida de este comando tendrán como prefijo el argumento "textured". Los más importantes son los de extensión obj, mtl y las imágenes png que corresponden a la textura del modelo. El archivo textured.obj puede visualizarse en Matlab [24], con lo que se pueden apreciar el buen resultado del procesamiento de las imágenes con *MVE* y *TexRecon* (Figura 9(b)). El modelo 3D puede visualizarse en línea en https://skfb.ly/o7rXH.

Figura 9. Resultados. (a) Fotografías originales del *dataset der_hass_20140923*. (b) Modelo 3Dcreado con *MVE* y *TexRecon*. Puede acceder a este modelo en https://skfb.ly/o7rXH.

5. CONCLUSIÓN

El uso de *Docker* para ejecutar programas nativos de un determinado sistema operativo se ha popularizado en los últimos años. La facilidad para instalarlo y utilizarlo se está convirtiendo rápidamente en una alternativa eficiente para ejecutarcualquier aplicación. No obstante, para usuarios poco experimentados en *Linux*, puede parecer complicado hacer funcionar aplicaciones desde cero. El uso de contenedores *Docker* evita la emulación de un sistema operativo completo de manera tradicional. En este artículo se han mostrado los pasos que deben seguirsepara la instalación y compilación de un programa específico, *Multi-ViewEnvironment*. El

objetivo principal es mostrar que no se requiere de un equipo muyespecializado para realizar fotogrametría y generar modelos tridimensionales de objetos físicos. *MVE* es un programa que procesa una gran cantidad de imágenes,por lo que suele ser muy voraz y pone a prueba el manejo de recursos de los contenedores *Docker*. Los resultados de la modelación son comparables a los quepueden obtener con equipos de fotogrametría comerciales.

6. AGRADECIMIENTOS

"Los autores agradecen al Laboratorio Nacional de Supercómputo del Sureste de México (LNS), perteneciente al padrón de laboratorios nacionales CONACYT, por los recursos computacionales, el apoyo y la asistencia técnica brindados, a través del proyecto No. 202003039N."

"The authors thankfully acknowledge computer resources, technical advice and support provided by Laboratorio Nacional de Supercómputo del Sureste de México(LNS), a member of the CONACYT national laboratories, with project No. 202003039N."

REFERENCIAS

[1] Photomodeler Technologies, *Photogrammetry, 3D measurements from Photos,* PhotoModeler Technologies. En línea en https://www.photomodeler.com/. Último acceso: 23 de junio de 2019.
[2] Hexagon AB, *Photogrammetry Software*. En línea en https://www.hexagongeospatial.com/products/power-portfolio/imagine-photogrammetry. Último acceso: 15 de noviembre de 2019.
[3] Bentley Systems, Incorporated, *Context Capture.* En línea en https://www.bentley.com/en/products/brands/contextcapture. Último acceso: 25 de octubre de 2019.
[4] Agisoft, *Metashape. Photogrammetric processing of digital images and 3D spatial datageneration*. En línea en https://www.agisoft.com/. Último acceso: 15 de diciembre de 2020.
[5] 3DF Zephyr, *The Complete Photogrammetry Solution*. En línea en https://www.3dflow.net/3df-zephyr-photogrammetry-software/. Último acceso: 12 de diciembre de 2020.
[6] Capturing Reality S.R.O., *RealityCapture: Mapping and 3D Modeling Photogrammetry Software*. En línea en https://www.capturingreality.com/. Último acceso: 21 de marzo de 2020.
[7] J. L. Schonberger y J.-M. Frahm, *Structure from Motion Revisited* de *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, Nevada, United States, 2016.
[8] J. L. Schonberger, E. Zheng, M. Pollefeys y J.-M. Frahm, *Pixelwise View Selection for Unstructured Multi-View Stereo*. En *European Conference on Computer Vision (ECCV)*,Amsterdam, Netherlands, 2016.
[9] Alice Vision, *Photogrammetric Computer Vision Framework.* En línea en https://alicevision.org/. Último acceso: 01 de diciembre de 2019.
[10] E. Rupnik, M. Daakir y D. Pierrot, *M. MicMac A free, open-source solution forphotogrammetry. Open geospatial data, software. stand.,* vol. 2, nº 14, 2017.
[11] Fuhrmann et. al. *MVE - A Multi-View Reconstruction Environment* de *Eurographics Workshop on Graphics and Cultural Heritage*, Darmstadt, Germany, 2014.
[12] S. Andrew, *Build Instructions for Windows*. En línea en https://github.com/simonfuhrmann/mve/wiki/Build-Instructions-for-Windows. Último acceso: 15 de junio de 2021.
[13] Technical University of Darmstadt, *TexRecon, Graphics, Capture and Massively parallelcomputing.* En línea en https://www.gcc.tu-darmstadt.de/home/proj/texrecon/index.en.jsp. Último acceso: 19 de julio de 2020.
[14] Docker, *Docker, Empowering App Development for Developers*. En línea en https://www.docker.com/. Último acceso: 16 de mayo de 2019.
[15] Oracle, *¿Qué es Docker?* En línea en https://www.oracle.com/mx/cloud-native/container-registry/what-is-docker/#docker-

image. Último acceso: 17 de diciembre de 2020.

[16] Canonical Ltd. Ubuntu, *The Linux command line for beginners*. En línea en https://ubuntu.com/tutorials/command-line-for-beginners#1-overview. Último acceso: 29 de junio de 2020.

[17] Indepent JPEG Group, *Libjpeg*. En línea en http://www.ijg.org/. Último acceso: 19 de noviembre de 2019.

[18] Greg Roelofs. contributing authors. *Libpng home page.* En línea en http://www.libpng.org/pub/png/libpng.html. Último acceso: 15 de noviembre de 2019.

[19] S. Leffler, *Libtiff, TIFF Library and Utilities*. En línea en http://www.libtiff.org/. Último acceso: 15 de noviembre de 2019.

[20] The Khronos Group Inc., *OpenGL, The Industry Standar for High Performance Graphics*. En línea en https://www.opengl.org/. Último acceso: 18 de septiembre de 2020.

[21] M. a. M. N. a. G. M. Waechter, *Let There Be Color! Large-Scale Texturing of 3D Reconstructions*, de *ECCV 2014*, Zurich, Switzerland, Springer International Publishing, 2014,pp. 836-850.

[22] Technical University of Darmstadt, *Multi-View Environment- Graphics, Capture and MassivelyParallel Computing.* En línea en https://www.gcc.tu-darmstadt.de/home/proj/mve/. Último acceso: 25 de noviembre de 2020.

[23] P. Cignoni, M. Callieri, M. Corsini, A. M. Dellepiane, F. Novelli y G. Ranzuglia, *MeshLab: anOpen-Source Mesh Processing Tool,* de *Sixth Eurographics Italian Chapter Conference*, 2008.

Correo de Autor: ***rdc06607@docente.ujat.mx***