Instalación del Simulador HALCON

Ing. Leonardo Luis Ortiz

Instituto Balseiro
Centro Atómico Bariloche

28 de julio de 2025

Contenido

Instituto Balseiro

HALCON

DSP Simulation Engine

Instituto Balseiro

HALCON

HALCON es un proyecto de código abierto desarrollado por Patricio Reus Merlo que consiste en un conjunto de herramientas desarrolladas en C++20 y Python3 que permiten a los usuarios diseñar y verificar arquitecturas de procesamiento digital de señales (DSP) desde las primeras etapas de diseño (alto nivel) hasta la implementación digital (RTL).

Instituto Balseiro

HALCON - Instalación

Los requerimientos más importantes son Docker y Git. Obviamente es necesario tener instalado un editor de texto y en el caso de utilizar Windows, también hay que instalar WSL.

1. Instalar **WSL**:
   - Ubuntu: no hace falta
   - Windows
2. Instalar **Git**:
   - Ubuntu: no hace falta
   - Windows: aunque es opcional, se puede hacer desde WSL
3. Instalar **Docker**:
   - Ubuntu:

```
1  curl -fsSL https://get.docker.com -o get-docker.sh
2  sudo sh get-docker.sh
3  sudo docker run hello-world
```

   - Windows: también hay que activar el uso de Docker en WSL

Instituto Balseiro

HALCON - Instalación

Es importante señalar que correr los simuladores en Windows con WSL es mucho más lento que correr los simuladores en Ubuntu nativo. Por lo tanto, se recomienda Ubuntu nativo para simulaciones largas o barridos de parámetros.
Después de la instalación de Docker, verificar que está funcionando con:

```
docker --version
```

Importante: en Windows hay que tener corriendo Docker Desktop para que WSL lo levante. El comando para verificar que Docker está funcionando, y todos los comandos de aquí en adelante, se corren en una terminal WSL2 de Ubuntu (o cualquier otra distro de Linux). Para definir WSL2 por defecto ejecutar el siguiente comando y reiniciar la computadora:

```
wsl --set-default-version 2
```

Instituto Balseiro

¿Qué repositorios clonar?

No es necesario clonar todos los repositorios ya que están enlazados entre sí mediante sub-módulos de Git. Es decir, cada repositorio incluye a los repositorios que requiere para funcionar.
Para este tutorial, es suficiente con el repositorio Examples:

```
cd <some_path>
mkdir halcon
cd halcon/
git clone https://gitlab.com/hawk-dsp/examples.git
cd examples/
git submodule update --init --recursive
```

Instituto Balseiro

HALCON - Instalación

Sin embargo, si se desea clonar los principales repositorios de HALCON para participar en el desarrollo del proyecto, los comandos son los siguientes:

```
cd <some_path>
mkdir halcon
cd halcon/
git clone https://gitlab.com/hawk-dsp/core.git
git clone https://gitlab.com/hawk-dsp/release.git
git clone https://gitlab.com/hawk-dsp/modules.git
git clone https://gitlab.com/hawk-dsp/examples.git
git clone https://gitlab.com/hawk-dsp/tools.git
git clone https://gitlab.com/hawk-dsp/tutorials.git
cd examples/
git submodule update --init --recursive
cd ../tutorials/
git submodule update --init --recursive
```

Instituto Balseiro

HALCON - Instalación

Importante: con el fin de simplificar las cosas, de aquí en adelante y a lo largo de todos los tutoriales se va a hacer referencia a los directorios del proyecto con las siguientes macros:

- HALCON = <some_path>/halcon
- CORE = <some_path>/halcon/core
- RELEASE = <some_path>/halcon/release
- MODULES = <some_path>/halcon/modules
- EXAMPLES = <some_path>/halcon/examples
- TOOLS = <some_path>/halcon/tools

Por ejemplo, el comando cd HALCON implica cambiar al directorio <some_path>/halcon.

Instituto Balseiro

¿Cómo correr el contenedor?

Antes de lanzar el contenedor hay que ubicar al archivo dockerfile en el repositorio y construir el contenedor. La construcción se hace una única vez a menos que se hagan modificaciones en el dockerfile, en tal caso hay que volver a construirlo.

1. Ubicar el archivo dockerfile: si clonaste Tools, el dockerfile está en HALCON/tools/docker. Si solo clonaste Examples, el archivo está en HALCON/examples/lib/tools/docker. Para no perder generalidad en el tutorial, nos referiremos a este path como DOCKER.
2. Construir el contenedor (tarda unos 30 minutos):

```
cd {DOCKER}
sh build.sh
```

Importante: el contenedor siempre se debe construir en el directorio DOCKER porque es donde se encuentra el archivo dockerfile.

Instituto Balseiro

3. Lanzar el contenedor

Importante: el contenedor siempre se debe lanzar en el directorio HALCON. De este modo, siempre se mapea el directorio /app del contenedor con el directorio

Instituto Balseiro

HALCON - Instalación - Algunos comentarios

1. De aquí en adelante todos los comandos se ejecutan dentro del contenedor a menos que se indique lo contrario.
2. El archivo dockerfile describe todo lo que el contenedor tiene instalado por defecto con sus respectivas versiones. Por ejemplo, el contenedor de este proyecto ya tiene Git, Python3, Doxygen y gcc13. Además, tiene algunas librerías de C++ y algunos módulos de Python.
3. Todo lo que se instale dentro del contenedor (y no a través del dockerfile) se pierde al cerrar el mismo. Por lo tanto, en caso de necesitar instalar algo, hacerlo a través del dockerfile y volver a construir el contenedor.
4. Es una buena práctica ser explicito con las versiones de las librerías de las que depende el proyecto (más que todo en C++). Por lo tanto, siempre poner las versiones de lo que se instale en el dockerfile porque descubrir que un bug se debe al cambio de versión de una dependencia es una tarea titánica.

Instituto Balseiro

¿Cómo correr un simulador?

En el tutorial del proyecto HALCON en gitlab están detallados los pasos para correr el simulador de forma manual. En nuestro caso haremos uso del script main.py que automatiza el proceso de compilación, ejecución del simulador y ejecución de tests en una sola línea:

```
cd {EXAMPLES}/low_pass_filter_sim/
python3 main.py -h          # Lista opciones (opcional)
python3 main.py -f -r -t    # Compila (-f), ejecuta (-r) y corre
    plot.py (-t)
```

Importante: el script main.py tiene dos flags de compilación diferentes, -p y -f. La primera opción compila el proyecto solo simulando los flancos positivos de los clocks mientras que la segunda compila una versión donde ambos flancos son simulados. Si la aplicación no demanda simular los flancos negativos de los relojes, se recomienda compilar con -p ya que en este modo el tiempo de ejecución del binario es la mitad que para el caso compilado con -f.

Instituto Balseiro

Q&A

Instituto Balseiro