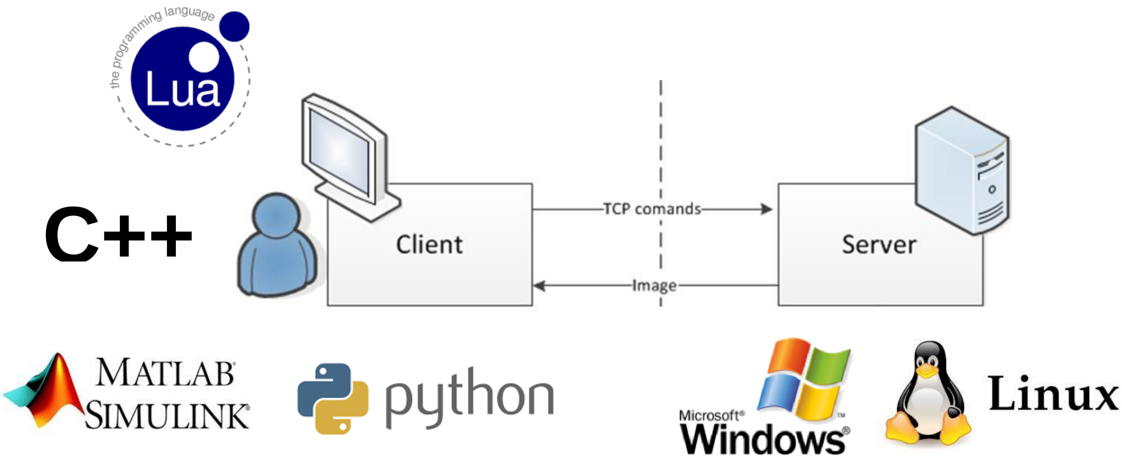
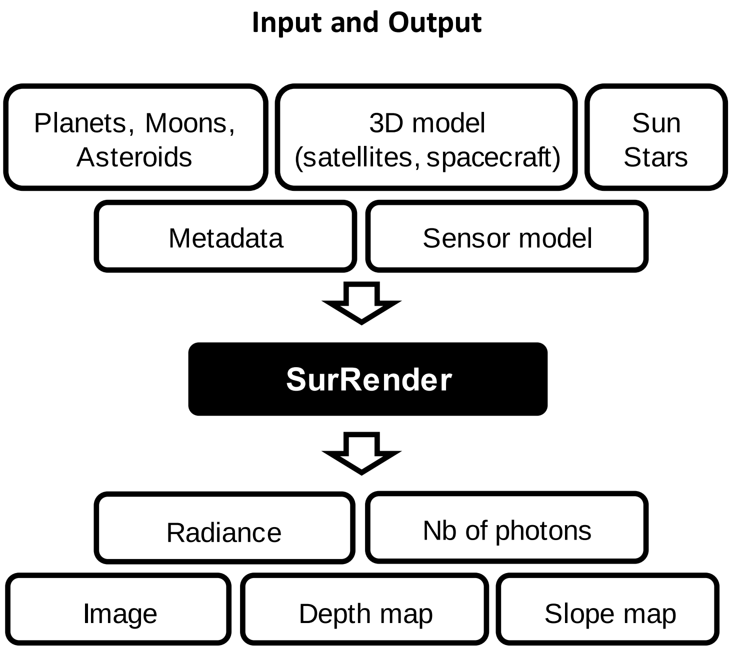
SurRender software
Technical features

The current version of the software is SurRender 6.0. The SurRender software is used through a client-server mode: SurRender's main application runs a server. The SurRender server can be located on the same computer as the client, or on a remote computer (possibly a cloud). The server receives commands from the client trough a TCP/IP link, and sends the resulting image back.

The SurRender client is provided to the user as a unified API for many languages. Especially, one can call SurRender from Lua, C++11, MATLAB (with or without Simulink) or Python 3.



SurRender supports a great variety of scene configurations. A schematic of the scene components is shown in the next figure. SurRender capabilities are presented in more details in a dedicated article available online. In the table below, we summarize SurRender main technical features.



Software	- Linux or Windows - Client-server protocol (TCP/IP) - Multiple interfaces: Python 3, MATLAB / Simulink, C++, Lua - Hardware in the loop - Cloud computing
Rendering	- Raytracing (CPU): physically accurate - OpenGL (GPU): real-time - Highly efficient simulation of sparse scenes (space environment) - Highly optimized to handle a wide range of resolutions and detail levels *The raytracer implements the physical principles of light propagation*
Supported data	- Digital Elevation Model (DEM), Textures, albedo maps - JPG, TIF, PNG, … - NASA PDS data format - Giant textures (up to 256 TB) - Procedural texture/DEM generator (fractal) - 3D meshes (OBJ, 3DS, PLY, Collada, …) *SurRender handles data at Solar System scales, from millions of kilometers to sub-meter distances*
Objects	- Artificial objects: 3D mesh models (satellites, robotic spacecrafts) - Planets, asteroids, moons: analytical shapes - Very high-res planet-wide DEMs - Sun, star background - Custom shapes *Solar System objects are stored in memory much more efficiently than meshes*
Image acquisition	- Images rendered in physical units (W/m^2) - Slope maps, depth maps - Various projection models & distortions (pinhole, fisheye, orthographic) - PSF, variable PSF in the FOV. - Achromatism, defocus - Acquisition modes (sampling): global shutter, push-broom, snapshot - Windowing (variable integration time in the pixels) *The raytracer design simulates the sensors working principle at physical level (photons)*
Models	- Embedded modelling language SuMoL (with dedicated editor) - Analytical or numerical models can be implemented at will - BRDF, projection, sensors, geometrical objects, etc.
Sensors	- Sensor models: generic sensor, HAS2, HAS3, JUICE/NAVCAM - Various effects: integration time, readout noise, photon noise, dark current, transmission, gain, quantum efficiency, pupil diameter, motion blur - Active sensors: LIDAR, time-of-flight cameras (with light spots, lasers) - Optical (B&W, RGB, multispectral) or infrared (no thermal model yet) - Spectrum and bandwidth
BRDF	- Lambertian (mate surfaces) - Mirror - Hapke (Moon /asteroid surfaces, regolith) - Oren-Nayar (mate) - Phong (plastic, MLI) - Custom models
Notable effects	- Geometrically correct reflections & shadows - Eclipses, soft shadows - Blooming - Secondary illuminations - Continuous simulation over large distance range (from millions of km to sub-meter scales) - Simulates motions, including detector motions (push-broom, rolling-shutter, micro-vibrations, …) - Subpixel accuracy

