Craft Director Studio™ - Whitepaper

Version 1.1

This paper is a technical description of Craft Director Studio™ and the technology behind it. First developed in 2003, Craft Director Studio is a cutting edge software solution for generating realistic movement. This paper describes the technology that makes this possible, the architectural make-up behind Craft Director Studio, and how it can be extended even further using its built-in API.

1. Introduction

The idea of Craft Director Studio™ (CDS) emerged in 2003, when Craft Animations founder Luigi Tramontana realized the ever-tedious work behind 3D-animated sequences created for any purpose. Today, whenCG visualization technology is nearing the point of absolute photorealism of computer generated content, we still use the same kind of animation technology as Walt Disney did back in the 1940's. By completing each picture in the sequence, frame by frame, you are able to create motion for your characters. This is no different from today. Even though the process has been digitized and the objects can simply be moved instead of needing to be completely redrawn, each picture in the sequence still needs to be manually modified frame by frame.

What Tramontana realized is that this process is not only time-consuming; it also very commonly produces insufficient results. Some motion is simply too bound to its natural laws of movement, and it is very hard to replicate this movement by hand.

Some introductions of technology have indeed emerged in the animation category – motion capture being the most widely used technology of creating realistic motion, primarily for characters. But the technology is limited to its source and advanced equipment, and everything is a result of mere interpolation, no generation.

This is where CDS emerges. It mimics the motion behaviour of specific objects, generating an animation sequence with the same intricate level of detail as if the motion was captured. This paper describes the technology allowing this to happen.

2. Workflow

The control of the tools in CDS is designed to mimic the control of the objects in the real world. For example, to control a car, the animator simply steers it. For vehicles, this is usually achieved in the same manner as in computer games where you control the vehicle with a gamepad or joystick.

As a result, keyframing is performed automatically, so the only thing the animator has to focus on is how the end result turns out. The recording process time is the same time it takes to play the animation back, giving the animator plenty of time to adjust or re-record the animation as opposed to using traditional keyframing methods.
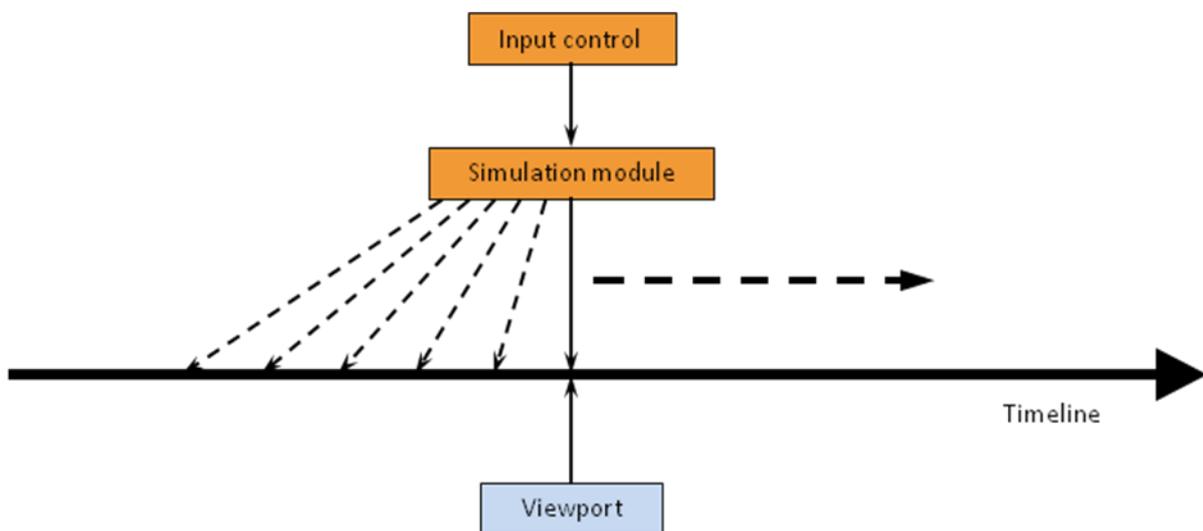
Figure I.   The simulator module manages the keyframing and the viewport is used for visual real-time feedback.

2.1. Recording routine

CDS works in a bi-state environment. The default is the passive mode, where the host application has control, the user can create and modify meshes, tools, settings, etc. just as usual. The animator enters the active mode by pressing the record button in the CDS interface. At this state, CDS takes control over the host application and turns the environment into a virtual environment. The animator becomes the driver, and all movement in the environment is automatically keyframed while the viewport is simply used as a visual feedback to the animator. CDS goes back to passive mode once the stop button is pressed.

2.2. Iterations

When using CDS to animate a car, for example, the animator faces the same conditions as the stunt driver does. Every act has its consequence and every movement is recorded. The difference is that the stunt

driver cannot afford mistakes, but the animator can. The animator can simply reset the timeline and record the animation once again. This is the main method used to achieve a specific task or motion.

2.3. Slow motion

In some cases specific maneuvers may be required which simply is too hard to control in real-time. For such cases, CDS provides the means of recording the animation in reduced tempo, making even very complex maneuvers much easier to achieve.

2.4. Seamless transitions

This is the key technology that makes this kind of workflow possible. At any point in the animation, the animator can revert and reiterate the recording while the resulting animation is always completely seamless, just as if the animator completed the recording in one session.

2.5. Host symbiosis

CDS works in complete symbiosis with the host application and does little interference with traditional workflow for the animator. There is no need to import or export information to foreign data structures or to rely on an external application. Once a recording is performed, the result is directly assembled into the internal animation system and can be manipulated as any keyframed animation.

2.6. Control

The most common problem with the CDS workflow is the feeling of lack of control. Since everything is generated in real-time the animator is bound to make mistakes. The iterative workflow is designed to remedy exactly this. The animator achieves control by quantity.

3. Platform

3.1. Supported operating systems

CDS uses Qt [http://qt.nokia.com] a platform-independent GUI library which enables CDS to operate independent of operating systems. In addition, CDS uses a licensing system provided by Reprise Software [http://www.reprisesoftware.com]. Its product, Reprise RLM also supports a variety of different platforms and operating systems.

Today, CDS supports Microsoft Windows operating systems with x86 and x64 architecture, Mac OS X with i386 and x86_64 architecture, and Linux Fedora Core x64 distribution.

3.2. Supported host applications

CDS is not a standalone product, but runs as an application extension (plug-in) for specific host applications. In order for CDS to support a specific host application there are some criteria that need to be filled. Please see section 6.3 for more details.
To current date, following are the supported host applications to Craft Director Studio™:

- Windows platform
    - Autodesk Maya 7.0, 8.0, 8.5, 2008, 2009, 2010, 2011 32-bit
    - Autodesk Maya 8.0, 8.5, 2008, 2009, 2010, 2011 64-bit
    - Autodesk 3ds max 8, 9, 2008, 2009, 2010, 2010 design, 2011, 2011 design 32-bit
    - Autodesk 3ds max 9, 2008, 2009, 2010, 2010 design, 2011, 2011 design 64-bit
    - MAXON CINEMA 4D R10, R11, R11.5 32-bit and 64-bit
    - Autodesk Softimage 7.5, 2010, 2011 32-bit and 64-bit.
- Mac OS X platform
    - Autodesk Maya 8.5, 2008, 2009, 2010, 2011 32-bit
    - Autodesk Maya 2011 64-bit
    - MAXON CINEMA 4D R10, R10.5, R11, R11.5 32-bit
- Linux Fedora Core x64
    - Autodesk Maya 2008, 2009, 2010, 2011 64-bit

4. Content and Tools

CDS consists of roughly 30 motion characteristic modules. These solutions can be placed in four categories. For a complete list of currently provided tools, please see appendix A.

4.1. Vehicles

Vehicle modules are designed to mimic the motion of rigid body vehicles, such as cars, helicopters and airplanes, but also more complex vehicles such as tanks and other vehicles on tracks, semis and double-winged helicopters.

4.2. Cameras

Craft Camera Tools can generate advanced camera movements with very little effort. Long gone are smooth spline movements and artificially perfect shots.

4.3. Accessory and Utility tools

These modules are abstract tools which usually manage a very specific movement, but combined together they can perform very complex behaviours. For example, these tools can be used to add multi-wheeler support, movement of tank turrets and other accessories, and so on.

4.4. Pre-rigged models

Normally a specific model needs to be rigged with CDS tools before it can be animated with CDS. Although this only takes a few minutes normally, we suggest that our customers purchase models which are already rigged with CDS and can be animated instantly. Craft Animations is working with several modelling studios, (including but not limited to PerspectX and Dosch Design) that integrate pre-rigged models for the use with CDS.

5.   Architecture

CDS consists of a collection of modules, some used as abstraction layers and some as content providers. The modules communicate via different APIs, with the Core module lying as the central hub. All APIs and modules are developed in C++.
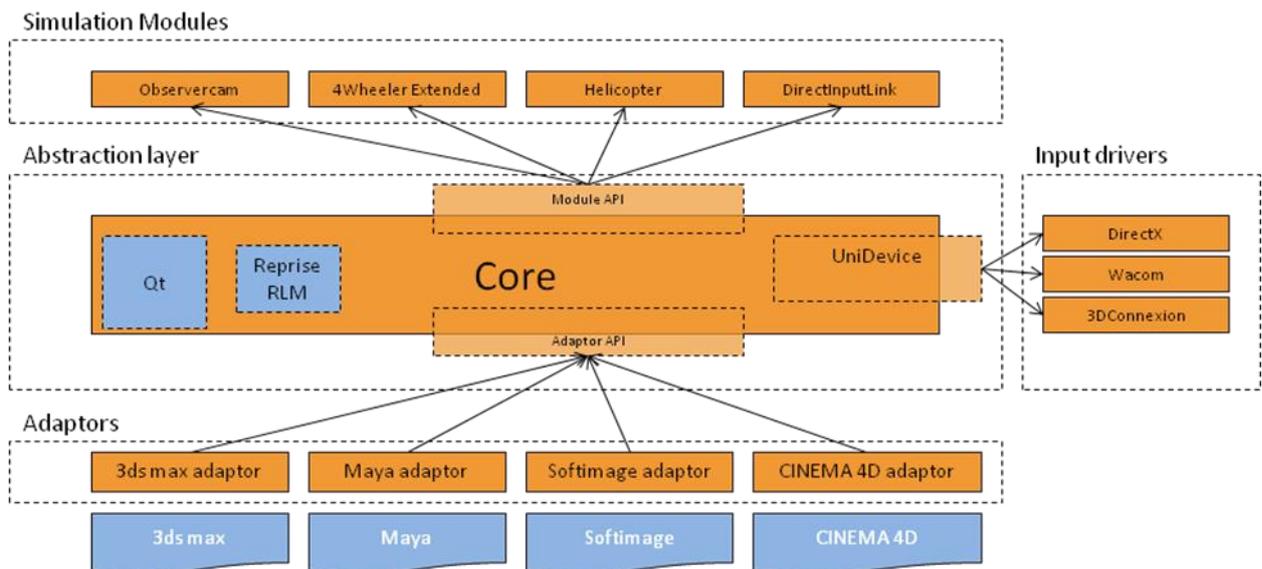
Figure II.   Modules contained in Craft Director Studio™

5.1. Core

This is the main module of CDS and contains the core functionality such as license, tool and input management. It provides the GUI interface for configuration and setup. It also manages all other modules contained in CDS.

5.2. Simulation Modules

Each tool is implemented in a simulator module, connected to Core via the module API. The simulator module can be implemented either as a dynamic library or as a static library linked into Core. The modules provide implementation for initializing the tool, creating the necessary objects for the tool, and registering the required input and configuration parameters to control the tool.

5.3. UniDevice

UniDevice is an internal module within Core, working as an abstraction layer for input devices. This module also manages the input driver modules.

5.4. Input drivers

The modules provide CDS with means of input to control the various tools. The most common input device is either a joystick or gamepad. On the Windows platform, this is provided by a DirectX input driver module, which basically translates the DirectX input information to the abstract UniDevice API. There are also other input drivers present, giving support for other types of input devices such as 3DConnexion mice and Wacom tablets.

5.5. Adaptors

An adaptor module functions as an intermediate layer between Core and a host application. Its purpose is to translate the signals between the host application API and Core API, and vice versa. In general, when CDS needs to extend its support to another host application or version, the only thing that needs to be developed is the adaptor to function as the link.

6. API and extendibility

The design of CDS was built to provide a high level of portability and easy expansion. This enables third party software and hardware vendors to add functionality and support to other host applications, or even use CDS as a middleware solution. Currently the APIs in CDS are closed for internal use only, but this will change in the future.

6.1. Module API

The module API is designed to extend CDS with tools to generate animated motion. The API is in C++ and requires no additional dependencies or libraries other than the CDS module SDK.

6.1.1. Node interaction

The sole purpose for the modules is to make objects in the scene move. Therefore the modules have extensive access to nodes owned by it, as well as limited access to nodes related to its own nodes. The nodes can be used both as input, as the movement of a node often depends of the state of the node in an earlier timeframe. The module can read the state of the node at any given time.

6.1.2. Input

In addition to that the nodes can be used as an input source; the modules can register a set of input controls which the user can bind to an input device, and also build a simple configuration window to enable the animator to adjust important parameters.

6.1.3. User interaction

Whenever the user does something in the scene that affects the tool, a callback is sent by Core to the tool telling what has happened. Some events are managed by Core itself to simplify the implementation of the tools. For example, if a user tries to delete a portion of the nodes that a tool consists of (for example, the front wheels of the 4Wheeler), Core will automatically interfere and delete all nodes belonging to the tool. This insures that the tool can always rely on that all dependent nodes are still present in the scene if alive.

6.2. UniDevice API

The UniDevice API is designed to extend the input control capabilities to CDS. The API is written in C++ and does not require any additional dependencies other than the CDS SDK.

The API accepts devices wrapped in a specific device class, which the module needs to implement. This class is then read and managed by UniDevice and serves as a link between CDS and the input module.

6.2.1. Module initialization

When the module is initialized, it registers the name and id of the driver type. It then registers which devices are available by the module. For instance, if two joysticks are available, the DirectX module would register two devices.

6.2.2. State poll

UniDevice works as a polling unit. Before the recording process starts, it notifies all drivers to initialize and prepare themselves.

On each frame, a poll function is called on all drivers where each driver should update the state on all devices. The state is then read by the tools to determine how they should behave.

6.3. Adaptor API

The Adaptor API is what the adaptor uses to communicate with Core. The adaptor usually needs to communicate with two APIs, the Adaptor API and the API provided by the host application.
In addition, the adaptor needs to implement a few template classes used by Core to notify the adaptor on certain events.

6.3.1. Initialization

The adaptor is responsible to load Core, either as a dynamic library or linked as a static library, and then calls the initialization routines.

6.3.2. Node management

The adaptor is responsible for managing the nodes being requested by the tools in Core. It also needs to make sure the nodes are properly represented in the host application.

6.3.3. Undo management

Undo management is one of the most technical challenges when porting to another host application. Core has an internal undo management system, and the adaptor must link it against the host application undo system. This is very crucial to make CDS fully integrated into the host application.

6.3.4. Key management

The adaptor must manage and translate all key information provided by Core to fit the keyframing system present in host adaptor.

7.   Technology

The design and interface of CDS is purposed to be simple and easy to implement. However, the power of CDS lies in its pockets of highly advanced recording routines and physical behavior. This is a product which has been under development for several years until it was initially released. Development has continued since then, with upgrades and adjustments.

Craft Animations has been working closely with universities to develop and test new technologies for evaluation to fit with CDS. Some have been incorporated, some have been discarded, and some are still to be embraced by CDS.

7.1. Platform independence

CDS is a platform-independent project where all platform-dependent source code is wrapped in an abstract layer for easy porting to not-yet-supported platforms.

7.2. Automatic control

Some vehicles, for example a helicopter, are usually very advanced and hard to control. To avoid forcing the animator to possess a helicopter certificate in order to animate it, a great part of the development of the tools is to simplify the control over them. Making the physical behavior of a specific vehicle is usually the easy part; it's being able to control it is what makes it hard. This is what CDS specializes in.

7.3. Abstract module API

Each tool in CDS is built in separate modules. These modules communicate using an abstract module API which effectively removes any dependency to the underlying architecture. The modules are therefore very easy to fit in other environments as well, as long as the abstract module API is implemented.

7.4. UniDevice

To unify the behavior of input signals to the modules, an abstraction layer for input devices was created. This layer enables the tools to receive input from a wide variety of input devices, regardless of type.

7.5. Host application independence

CDS is designed to support any kind of host application, as long as the host application fulfills required criteria. To achieve this, CDS provides an abstract host application API to the core functionality. This API

is then used within the implementation of a host-specific plug-in, an adaptor, and functions as a layer between the CDS core functionality and the host application.

7.6. Key and recording management

On a typical vehicle tool, hundreds of keys are generated each second during the recording state. Real-time response is essential for CDS and therefore it is very important that key management and recording process runs effectively. Most host applications are not designed to handle this kind of stress, and therefore CDS has to handle its own recording routine and management of keys. The keys are then batched into the host application for user access.

This management also handles the state of each tool and makes sure that each tool gets the ability to calculate the next step for the objects.

7.7. Customized physics routines

There have been attempts to use established physics engines within the tools to provide the physical properties of rigid body motion. However, the physics engines are normally too general to be able to provide the required level of realism for the vehicles. Therefore most tools are built using very custom-made physics routines, giving the user better experience and results.

7.8. External libraries

7.8.1. Qt
Qt is a platform-independent Graphical User Interface and is widely spread throughout platform-independent applications. This library has replaced an earlier proprietary solution because of its flexibility and friendly interface.

7.8.2. Reprise RLM
This library handles license management for CDS. It is a minimalistic library but provides powerful copy protection using encrypted signatures within the license files. It is very similar to FlexLM, but runs in a much smaller scale.

7.8.3. Bullet
Attempts have been made to try to incorporate a general physics engine into CDS for better interaction between the tools. This project has, however, been suspended as it does not provide any additional benefits for the animator. Physics engine collisions are generally too incorrect to be used as a high-end result. There are, however, some uses for a general physics engine at some instances. Currently, the only tool that uses Bullet as the underlying system is Crawler Tracks, which uses it to manage the track segments.

7.9. Master thesis

Following is a list of some of the master thesis being completed during the research related to Craft Director Studio™.

7.9.1. Mesh optimization and volume generation

By Ragnar Hammarqvist, master thesis 2008, Linköping University of Sweden

7.9.2. Controller for the simulation and filtering of human arm motion

By Tomas Björklund, master thesis 2007, Chalmers University of Technology, Sweden.

7.9.3. Craft Physics Interface

By Henrik Hansson, master thesis 2007, Linköping University of Sweden.

7.9.4. Constrained symplectic integrators for Hamiltonian systems

By Niklas Jansson, master thesis 2008, Chalmers University of Technology, Sweden.

7.9.5. Physical simulation of a car in 3D environment

By Johan Brännström, master thesis 2007, Chalmers University of Technology, Sweden.

7.9.6. Implementation and modification of a cloth simulator for deformable shells

By Linus Hilding, master thesis 2009, Linköping University of Sweden.

7.9.7. Realistic rendering of bird feathers in realtime

By Henrik Edin, master thesis 2007, Linköping University of Sweden.

7.9.8. Simulation of collision deformations

By Alessio Quaglino, master thesis 2008, KTH Royal Institute of Technology, Sweden.

7.9.9. Porting a 3D-renderer plugin

By Jonas Alfredsson, master thesis 2008, Linköping University of Sweden.

7.9.10. All purpose full physics wheel class

By Emil Jones, master thesis 2009, University of Kalmar.

8. Future and ongoing development

CDS has been continuously improved since its release in March 2006 and will continue to do so in the future. This chapter describes what is currently in development for CDS and what plans there are for improvements in the near future.

8.1. SDK for third-party developers

The SDKs used in CDS will be tweaked and opened up for external use. This enables third-party developers to enhance CDS with:

- More support of input devices to control the CDS tools.
- Custom-built tools specific for a certain task.
- Support to additional public or in-house modeling software.

8.2. Middleware solution

The goal for this project is to make CDS even more abstract and modular, creating the possibility for customer to actually license the tools separately, and use them for whatever purpose in third-party applications. This enables our customers to utilize the realistic camera and vehicle movement capabilities in whatever software they desire.

8.3. Custom development

CDS is a versatile tool, however, it can never support all requests being received throughout the years. To remedy this, Craft Animations has opened a new branch to provide services and be able to create solutions on-demand. This includes both tweaking CDS to customer's specific needs, or to create a complete solution based on the technology used in CDS.

Following are some examples where Craft Animations can provide services and solutions:

- Data visualization – visualize simulated data values as an animation. Reconstruct airplane movements prior a crash, visualize deformation data, etc.
- Provide solutions for specific movement characteristics. Need to animate a specific rig currently not supported in CDS?
- Integrate Craft Director Studio technology into existing workflow.
- Integrate Craft Director Studio technology into existing applications.

9. Appendix A – List of tools

This is a list of the tools currently available in Craft Director Studio™.

9.1. Camera tools

9.1.1. AutoZoomCam

Craft AutoZoomCam keeps one designated object constant in size while the camera moves. The camera adapts the field of view automatically, in exact proportion to the distance between the camera and the target. The target can be set to follow a moving object. For example if the target is attached to a car that is moving away from the camera, Craft AutoZoomCam will change the zoom in exact proportion to the car's speed.

9.1.2. HumanizerCam

Craft HumanizerCam adds subtle movement to your existing, animated camera. This will give the animation a much more human feel. Craft HumanizerCam can be configured to add just a little bit of noise to give it that little bit of extra life, to a nervous, hand-held camera motion.

9.1.3. MotionFilterCam

Craft MotionFilterCam can be used for smoothing and bounce in vehicle camera rigs or third person views. If you configure it for smoothing it works like Craft SoftMotionCam. Or you can configure it for bouncing. Either a soft bounce, as if you would react filming with a hand-held camera when a car drives over a bump or a stiff bounce, perfect for mimicking the motion a camera would get being mounted on a metallic rig on a car.

9.1.4. MultiStateCam

Craft MultiStateCam has 10 user-defined positions that a camera can jump to at the touch of a button. The user can determine how quickly or slowly the camera moves between the pre-defined positions. As the camera goes from one camera to another it will also transition the camera settings, such as field of view and target distance, to match the settings of the camera it is heading for.

9.1.5. ObserverCam

Craft ObserverCam is our most intuitive camera. If you move your control forward, the camera moves forward, if you move the control up, the camera goes up and so on. Use a gamepad, joystick, 3d mouse (and more devices) to control the camera in real-time. Get absolute control of the movement and zoom of your camera.

9.1.6. SoftMotionCam

Craft SoftMotionCam stabilizes turbulent and unnatural movements of an animated camera, making the motion smooth and natural. Almost like a steady-cam for the 3d camera. Craft SoftMotionCam can also be used as a trailing camera for a moving object, smoothing out any too sudden movements making the animation a lot easier to watch.

9.1.7. SphereCam

Craft SphereCam is constantly pointed at the same point. The camera can pan around the point and look at it from any angle, zooming in and out, moving around it in a big sphere. The center of the sphere can be attached to either a moving or still object. Use an input device like a gamepad/joystick or 3d mouse to control the camera and give it a very realistic, human feel.

9.1.8. Spline Speed-Controller

With Craft Spline Speed-Controller you can create waypoints in your scene. These waypoints will tell Craft Spline Speed-Controller's Carriage to pause, stop, go a particular speed, etc. This is a very powerful alternative to working with regular spline curves because you're not working with percentages along the spline, you're working in world coordinates. So if you decide to either extend or shorten the spline, you do not have to go through the entire spline to find out the percentages for the new positions. Craft Spline Speed-Controller's waypoints will snap to the closest point on the spline so you can move the waypoints around incredibly fast and easy.

9.1.9. ZeroGCam

Craft ZeroGCam is like a satellite in zero G space. If you give it a push, it will continue with a constant velocity in that direction until you give it a new force. This creates a sense of weightless motion.

9.1.10. ZoomAndFocusCam

Craft ZoomAndFocusCam gives you the ability to create and control focus effects in real time. You can shift focus between two points in real time using an input device (gamepad, joystick, etc.). This can either be done gradually or suddenly.
The two points can be attached to moving objects. For example: attach the two points to two cars in a car chase and shift the focus between the two moving cars at the touch of a button.

9.2. Vehicle tools

9.2.1. 2-Wheeler Extended

Craft 2-Wheeler Extended is a simulator for 2-wheel vehicles. This animation tool makes it easy for the user to create realistic simulations for vehicles in motion. It can follow the ground and interact with bumps realistically. Craft 2-Wheeler Extended can either be controlled either directly with an input device or be set to follow a target (which in turn can be set to follow a spline path).

9.2.2. 4-Wheeler Extended

Craft 4-Wheeler Extended is a simulator for 4-wheel vehicles. This animation tool makes it easy for the user to create realistic simulations for vehicles in motion. It can follow the ground and interact with bumps realistically. Craft 4-Wheeler Extended can either be controller either directly with an input device or be set to follow a target (which in turn can be set to follow a spline path).

9.2.3. Airplane Extended

Craft Airplane Extended is a high precision real-time tool for animating airplanes. The vehicle is controlled with an input device directly through the host application. Craft Airplane Extended can be used to simulate movements for all types of vessels, from toy airplanes to jumbo jets. Craft Airplane Extended also has the ability to follow spline paths.

9.2.4. Crawler Extended

Craft Crawler Extended is a simulator for track-driven vehicles. It can follow the ground and interact with bumps realistically. Craft Crawler Extended can either be controller either directly with an input device or be set to follow a target (which in turn can be set to follow a spline path).

9.2.5. CrawlerTracks

Craft CrawlerTracks is a tool for recording the simulated movement of caterpillar/crawler tracks. The tracks react realistically to the movement of the vehicle it is attached to and bounce and sway accordingly.

9.2.6. Helicopter

Craft Helicopter is an animation tool for simulating the movement of helicopters. Control it in real-time with a gamepad or joystick. Craft Helicopter is fully customizable to fit any helicopter model, anything from a small RC helicopter to a big military helicopter.

9.2.7. Trailer

Attach the trailer's hook to a moving object and the trailer will follow it accurately. Craft Trailer's wheels will follow the ground and interact with bumps realistically.

9.3. Accessory tools

9.3.1. DirectInputLink

Control anything in your 3d environment directly with an input device. Craft DirectInputLink works great as simple rigging utility. Since you're controlling your objects with human motion in real-time, you're getting animation that looks incredibly lifelike.

9.3.2. FirePower

Craft Firepower is a tool for animating guns, cannons, etc. Fire it using your keyboard/gamepad and automatically get a realistic recoil effect. Craft Firepower will also automatically clone bullets for every shot fired. Add another Craft Firepower act as the mechanism that shoots out the empty casings.

9.3.3. Missile

Craft Missile simulates projectiles like missiles and rockets. From anything big like a space shuttle, to something small like fireworks, this tool enables you to create a realistic animation based on real-world physics.

9.4. Freeware tools

9.4.1. Airplane Free

Is a free lightweight version of Craft Airplane Extended.

9.4.2. ObserverCam Free

Is a free lightweight version of Craft ObserverCam Extended.

9.4.3. 4-Wheeler Free

Is a free lightweight version of Craft 4-Wheeler Extended.

9.4.4. Utilities

Craft Director Studio™ includes a series of small tools used to augment the capabilities of the commercial tools. Craft Utilities are free for use.