# sugarcubescanner.jpg

# SugarCube Project Quick Summary

SoundFit SugarCube Model Generation System

System Overview, Release 2 Beta 1

July 2014

Akien MacIain

## Introduction

This document is a broad description of the SoundFit SugarCube 3D Modeling System. The system is a high precision scanner which produces 3D models of parts of up to 4cm in size. Model accuracy target is 50 microns.

This system is comprised of a hardware component, and a number of software components. The system uses Autodesk's ReCap modeling engine to convert photographic images to a 3D model.

## Overview Of Components

The system is composed of hardware component, as well as 3 major and quite a few minor software components. The 3 primary software components are the Embedded software, which runs inside the SugarCube… Host software that runs on a MS Windows capable computer having internet access, and attached to the SugarCube via USB… And server components hosted by both SoundFit and Autodesk.

The current host software is intended to both assist testing and manufacturing; and to be a beta test platform for end users.

### Hardware

The SugarCube Scanner implements a robotic photo studio with highly controlled spatial and lighting conditions. The scanner hardware connects to a host computer for image capture and upload. Status is reported via an RGB status LED on the face of the scanner, as well as via the host software. Polar 2 axis control of camera position, lamp control, and status reporting functions are built into the hardware.

### Embedded Software

The embedded software runs on the Arduino microcontroller in the system. It's written in standard Arduino C++. It does all of the operating of the robot, providing the following services:

#### Command I/O

Receives commands, processes them, and returns replies over the Arduino virtual serial USB port. All commands and responses are via this serial line, and are human readable (including a brief help screen and more verbose modes).

#### Mechanical Operation

This includes rotation and elevation control. This new version accepts motor control commands in degrees, and smoothly ramps up and down the motor speed, dramatically reducing vibration compared to the early beta devices.

#### Lamp Control

This includes imaging lamp control and status lamp control. In this new version, the brightness of the green and blue LEDs is controllable. The red one is either off or full on, in part because of limitations on the Arduino (insufficient PWM channels) and in part because red is used to report errors, and if there’s an error, we really want the user’s attention.

#### Detection & Reporting

This includes detection of 12 volt power required for the motors, that the drawer is closed, that the arms are in the home position, and so on. The reset command performs self-diagnostics using each of these sensors. The results are also available as responses to various query commands.

#### SugarCube-Specific Persistent Data Storage

For each SugarCube manufactured, this allows us to store and retrieve the scanner ID, camera name, and camera parameters used in EXIF population and template generation, with room for additional per-scanner info (motor or lighting parameters, for example) as may be needed.

### Host Software

There are currently two variants of the the host software, as of this writing, a TestBed version, for use mainly within SoundFit and an MVP Beta version aimed at our first set of audiologist beta testers. All variants of the host software share a common core library, called the SugarCube API. The host software connects to the scanner and provides the user interface. The host software is written in C#. The core functionality of the software is as follows:

#### Configuration Retrieval

On startup, the software checks the local config file. Both variants of the software provide a UI for setting these values. Once the software connects to the SugarCube, it also asks the SugarCube for it’s Z data, which tells it the cube serial number and which camera to use.

#### Process Shooting String

The shooting string is one of the configuration items. It provides a script for the execution of motor, imaging, and lamp control. The host system processes this script, sending some commands to the Arduino and executing others itself.

#### Image Capture

The host system operates the camera directly over the USB connection using the Windows DirectShow interface (via .NET). This allows the selection of any DirectShow compatible camera and use at whatever resolution is specified. These parameters are part of the stored configuration information.

#### Vibration Detection

Two images are taken, with a very small interval between them. The images are compared and if the changes exceed a preconfigured value, it is assumed that there is still motion in the system and a third image is taken, compared with the second and so on. When the two images match, motion is assumed to be zero and the last image is saved. For zero vibration images, our testing shows that vibration management should be on and motion sensitivity set to 1.

#### Image Averaging

Because camera noise can be an issue, a preconfigured number of images from each position can be averaged together, reducing image noise and enhancing sharpness. This option is computationally expensive, our Audiologist clients won't be able to afford to average too many per shot. Hobbyist or other industrial users will have more latitude for this if they require it.

#### Sigmoidal Contrast Enhancement

Autodesk has indicated that contrast is extremely important. This allows a preconfigured amount of sigmoidal contrast enhancement to be applied.

#### Add EXIF Data

Applies the 35mm equivalent lens focal length to the image. The data pushed into the EXIF fields comes from the Z data on the Arduino. This improves modeling enormously.

#### Manifest Generation

The manifest contains a complete list of images, angles for each shot, configuration parameters from the Z data, sender, addressee, server directives (such as scaling or cropping on or off, cropping block size, etc.), and so on. It is used by the server software when processing the submission of the scan to Autodesk.

#### Upload Scan

Images and the manifest are uploaded using SFTP. In the TestBed, uploads happen in real time. In the MVP Beta, uploads can be processed in real time or they can be queued for background uploading. The background uploading is managed by another piece of software call the SugarCube Uploader. The production version will have some kind of upload dashboard.

#### Triggering The Modeling Process

A URL is called that tells the server the upload is complete, and that it should begin the modeling process. The URL template comes from the configuration information on the host, and has the upload directory appended to it.

#### Installation

As of now, installation is manual, but a Wix based installer is part of the next phase of development.

### Server Software

Once a scan is taken, a user can simply submit the images to Autodesk's 123DCatch themselves, or process the images using open source programs such as VisualSFM (<http://ccwu.me/vsfm/>) and MeshLab (<http://meshlab.sourceforge.net/>), or they can use our service. Our service provides single button processing for most users. Our service exists as a SFTP server and a web server. The images are uploaded to the server, and then the trigger URL kicks off the processing. The server software is written in PHP.

#### Receives Scan

The server is running a file transfer protocol (FTP) server 24x7x365. Files are written to the FTP server by the client host, which stores them on the server. After all files have been uploaded, the client sends a ”trigger URL", launching a program that accepts the image files and begins processing them into a 3D model. The receiving program accepts the files quickly, so that the PC client doesn’t need to wait while the 3D modeling process takes place. It then launches the “Modeling Program” which handles Template Generation and Transmission to Autodesk.

#### Triggering The Modeling Process

See above in the Host Software section called Triggering The Modeling Process for the URLs that trigger the process.

#### Template Generation

A “template” file is generated from the information in the manifest (shooting positions and Z data). Conversions are made from polar to cartesian coordinates which then provide coordinates and angles for each of the images to Autodesk. Field of view information is provided from the Z string and applied to the template. Autodesk uses these template files to improve modeling efficiency and accuracy, as well as to ensure proper scaling of the imaged objects, and cropping of the floors and walls, so that the returned 3D model is ONLY the target object. That cropping minimizes further post processing CAD work that is typically an annoyance for users of other scanners.

#### Transmit to Autodesk

The server software sends the image files, and generated template to Autodesk’s ReCap server, which converts the 2D images into a 3D model. This generally takes 10-15 minutes of elapsed time. This program also can email the user (sender) confirming receipt of the model and beginning of modeling.

#### Callback Triggered by Autodesk

On completion, Autodesk launches our callback program. An output directory is created. The model is downloaded, converted it to the output formats: OBJ, STL, 3DP; and written to the output directory. Next it creates database entries, and sends email notifications to the manufacturer (recipient) with links to these files, and the three links below.

#### Image Gallery

The server software creates an image gallery view of the original images, and that link is included in the email.

#### Web Viewer

A web viewer (requiring WebGL) is added to the output directory and email that allows the user to view the model in their browser.

#### Difference Table

A table is created showing the differences between the template and the model we got back. This has proven to be a quantifiable and accurate means for assessing model quality. We may eventually use it as an automated way to assess quality and alert the user if the model is not adequate.

#### Prepare & Send Email

Server software notifies people about processing events. Notification emails are sent to either a “Sender” (the User who initiates the scan) and/or a “Recipient” (these may be the same person). The first notification acknowledges receipt of a set of scan images. After modeling is complete, the callback program starts. If Autodesk modeling fails we can send a failure message out. If the model succeeds a success message is sent to the recipient, with links to the OBJ, STL, 3DP files that can be downloaded, as well as links to the WebGL viewer (and optionally to the image gallery).

### Other Components

There are some additional components, some of which aren't strictly speaking software components in the same sense, but are fundamental parts of the system, and are more a knowledge function than a mechanical one.

#### Status LED

The Alpha code and hardware didn't support much in the way of user feedback. This system has a status LED that provides information to the user about the functioning of the unit. There are 5 color codes to the data:

Red - The embedded software is running, but there is no external power (set by reset function)

Blue - Idle/Ready (set by reset function)

Green - Scanning (shooting string dependent)

Purple - Arm is blocked (set by reset function)

Yellow - Taking image (shooting string dependent)

#### Shooting String

The shooting string describes the positioning of the camera for each of the shots. It also includes image capture, pause, lamp control, and status reporting commands. It even supports a looping structure so you can "start at x and take an image every y degrees".

#### 3D Surround

Recommended by Autodesk. This is one that we've observed makes a huge difference in modeling quality. Our current beta units are using one made from stacks of pennies, wrapped in specific paper covers.

#### FlashCube

A utility for flashing new software to the SugarCube without needing the Arduino software. By “flashing” we mean storing embedded program data into the scanner.

#### SugarCube Arduino Smoke Tester

A utility for testing serial connectivity to the SugarCube.

#### Z Data Values

The "Z Data" are items related to specific scanners, such as serial number, camera name, and so on. These are retrieved from the scanner by the host software in order to know how to use the scanner. Some of this data is also used by the server side components to control modeling.

#### Focus Block

A Lego component for constructing a platform for focus testing.

#### Wall Surround

The surfaces of the inside of the device are covered in a design that is high contrast and non-repeating. This is a key part of information provided in the images. The wall surround can be found in Subversion.

### Templates

Templates are a special case of a model. The model files come back as .3dp files. These describe the model in Autodesk's database, the positions of the cameras within the "photoscene", and their attributes.

We have two methods for creating templates at this time. A manual process and an automated process. The software allows a photoscene to be submitted without a template (forcing Autodesk to calculate all the camera parameters and positions); to be submitted with a manually created template; or to be submitted with an automated template.

Which template and parameters a model will be submitted with at this time depend on options set in the host software, including the triggering URL. A later version will have only one URL and will have additional options in the software.

Autodesk tells us that the camera positions are starting places and that being exact isn't important.

Manual templates are made from models and are specific to a given shooting string. Ray has developed a documented process for creating these templates. If you change the positions in your shooting string, then you also have to update your template.

Automated templates are generated from the shooting positions given in the shooting string. Each one is created uniquely for that specific photoscene. They're created from software rules that enable us to fine tune all the templates as we learn more about the nuances of Autodesk's software.