

# Open source hardware automated guitar player

Andrew Henry

School of Computer Science  
University of Lincoln, UK  
14552894@students.lincoln.ac.uk

Charles Fox

School of Computer Science  
University of Lincoln, UK  
chfox@lincoln.ac.uk

## ABSTRACT

We present the first open source hardware (OSH) design and build of a physical robotic automated guitar player. Users' own instruments being different shapes and sizes, the system is designed to be used and/or modified to physically attach to a wide range of instruments. Design objectives include ease and low cost of build. Automation is split into three modules: the left-hand fretting, right-hand string picking, and right hand palm muting. Automation is performed using cheap electric linear solenoids. Software APIs are designed and implemented for both low level actuator control and high level music performance.

## 1. INTRODUCTION

Robotic musical instruments have four major use cases: As assistive technologies, they can enable disabled users to be part of musical performances through semi-automation, such as automating the work of one hand but leaving the user to play the other. Similar semi-automation of manual skills can be used in education to enable students of an instrument to focus on learning one hand at a time. As artistic installations, fully automated robotic musicians can be used to perform live music on physical instruments which is hard or impossible for human performers to play, but retaining the live physical quality of their acoustic instruments. As composition tools, they can be used by human composers who are not trained to perform on the instrument themselves but would like to write for it, including for high quality, high accuracy studio recordings without the need for session musicians.

The guitar is one of the world's most popular instruments and many studio composers would benefit from a perfect and free robotic session musician able to perform and record exactly what they specify. Guitars are also used for social playing in groups, where semi-automation could enable disabled and learning players to replace one hand and join in the music by playing with just the other.

We thus present the first open source hardware robotic automation system for guitar. It comprises modules for the left hand fretting (fig.1) and right hand string picking including strumming emulation, and a right-hand palm muting module. There are also two software components, a serial transceiver and an automation controller, used to interface the system to other music software. The key benefit of open source hardware is that is open for users from

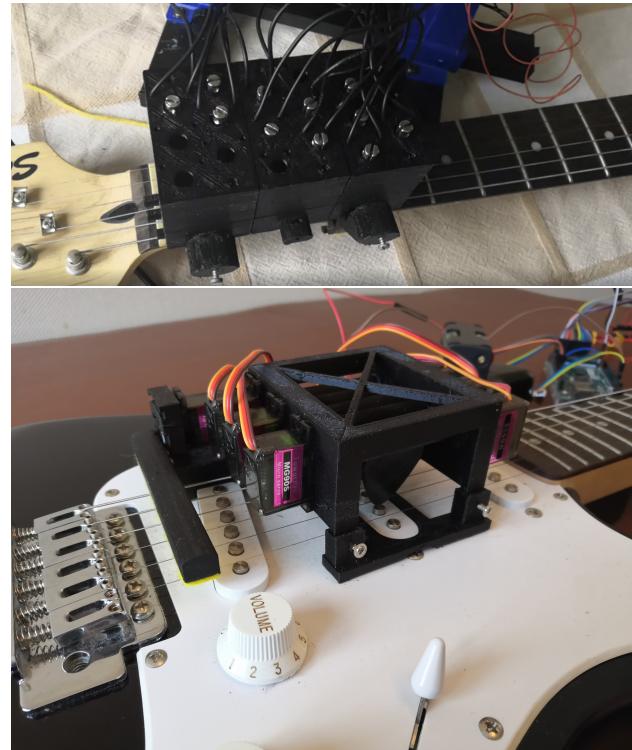


Figure 1. Photograph showing (above) the fretting module and (below) the picking and muting modules, mounted on a Stratocaster style guitar

different cultures to modify to extend with the additional techniques that are most important to them. OSH places particular requirements on design including the need for components to be easily and cheaply available, and easy and cheap to assemble by non-specialists, and we take account of these considerations. The robot player is able to play chords and melodies using either or both of the left and right hands, possibly with a human playing the other hand, or possibly both automated together.

### 1.1 Related work

The MIT guitar machine I [1] is a semi-automated augmentation to the human guitar player, allowing a skilled human and the machine both press down strings together to enable previously impossible combinations of notes to be played. The MIT guitar machine II [2] expanded on the real-time augmentation aspect, changing the focus from the fret fingering to the strumming to produce new sounds from the guitar using a combination of E-bow coils, solenoids and motorized picking actuators. These systems were intended to extend the sonic range of the instrument into a hyper-

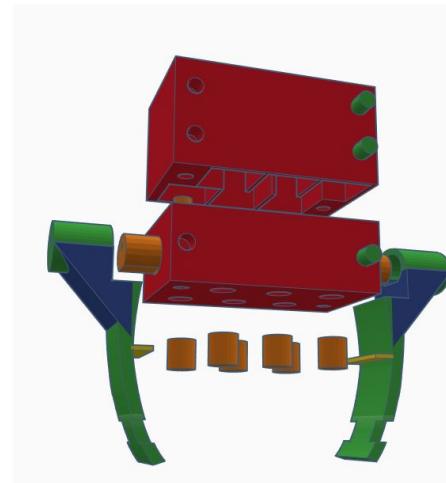
instrument, rather than to automate the regular instrument.

Crazy J [3] is capable of fully automated playing, using solenoids with levers to press down strings, but is not OSH. Strumbot [4] is a fully autonomous player. Rather than attaching to a regular guitar, it is built by heavily and irreversibly modifying a guitar, so it is not possible to remove the automation system from the instrument. To strum the strings a single horizontal double arm is used, allowing for good control of the end effector without having a brace in the middle of the arm while allowing the distance of the end effector to be changed. The end effector holds two picks so that StrumBot can quickly transition from up-strums to down-strums with minimal motion. Strumbot is fretless, using continuous motion carriages are used to change the pitch of the string like a cello, which enables it to simulate string bending and to play non-Western microtonal pitches, but at the cost of complex and bulky carriage hardware. Swivel [5] is a robotic guitar system that allowing precise control over dynamics and pitch content. A plucking mechanism consists of a stepper motor driving a wheel holding plectrums, spun round to strum the string. A servo adjusts the height of the subsystem to change the velocity of the plucking. A damping mechanism acts as a palm-mute device by lowering a felt pad on to the string to dampen the sound. Fretting uses an arm to press down on the fret, rotated using a stepper motor and raised and lowered via a solenoid. The arm has no way of moving forwards making it move in an arc, limiting the frets it can press. Lemur GuitarBot [6] has a simple design making it easier to understand and construct; due to non-modular design it would be hard to adapt for a normal guitar without damaging or changing the instrument.

Robotic guitar players have recently appeared as artistic and entertainment installations, such as Compressorhead [7] and Z-Machines [8]. But as proprietary installations they are not available for other composers to use. Similarly, some commercial semi-automation assistive and education systems have been developed, but focusing on particular styles of music. None of the above are OSH to enable musical community members to modify them for their own instruments, styles, and research.

## 2. DESIGN CONSIDERATIONS

OSH allows for more effective and accessible sharing and collaboration among researchers [9], as well as more accessible entry-level projects due to the ease of access of how to duplicate and repeat the project. OSH is a relatively new concept with competing definitions. *Deep OSH* is the strongest form, in which designs for all levels of components and all tools used in the project are public and not patented. *Weak OSH* is based on conventional Product Design and considers the design itself to be the only component in need of openness. The design can be completely dependent on patented components and tools. *Shallow OSH* allows designs to be built from patented *implementations* of component designs and tools on the condition that the *interface* to these components and tools is made public and not patented. This allows multiple suppliers to produce alternatives for required components, improving the access and availability to the required components. There is then a continuum from shallow to deep,



**Figure 2.** CAD model of the fretting module

in which the depth of an OSH design can increase as more sub-components are swapped for *Deeper OSH* designs, until a fully deep design is reached.

Our design is currently weak OSH, due to some sub-components (e.g. Arduino) being weak OSH. We hope that the design will become shallow then deeper in the future as shallow and deep alternatives to these sub-components become available.

Common to all OSH definitions are fundamental but vague requirements that designs should be cheap and easy to build. We interpret this to mean that both components and tools should be affordable by musicians and available from multiple online suppliers in most countries, and that the design should be buildable with high-school workshop skills.

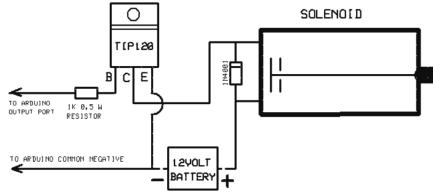
OSH communities are multicultural, and for musical systems these cultural differences are manifested as differences in instruments and performances styles. ‘The guitar’ is not a single instrument, and an OSH design should be either immediately usable, or easily modifiable to be usable, by as wide a collection of guitars and guitar-like instruments as possible. The present design is intended to work out of the box with most western electric and acoustic six string guitars. The OSH design is set up so it can easily be modified for use with other guitars such as 7-strings and basses. It is also modular and intended to be extendable, for example musicians using styles involving string bending could easily add a bending module. To varying degrees, the design could also be modified for use with other lute family instruments, with the work required roughly proportional to the instrument’s difference from a western guitar.

## 3. DESIGN

The following is an overview of the design. The full, buildable design, bill of materials, step-by-step build instructions, and demo codes and videos are available from [gitlab.com/Andrew\\_Henry/automated-guitar](https://gitlab.com/Andrew_Henry/automated-guitar)

### 3.1 Fretting module

The fretting module (fig. 2) is formed of three CAD objects: holding box, support legs and fingertips. The holding box is made to contain all the solenoids required for

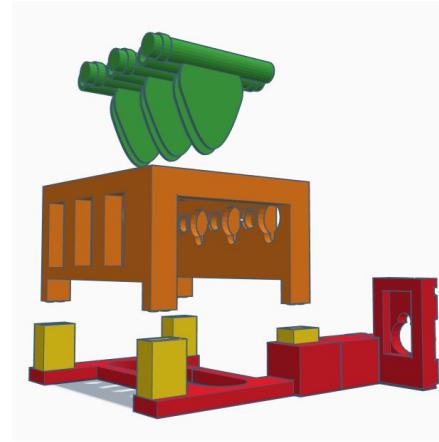


**Figure 3.** Circuit diagram for the fretting module

each fret and is split in two, allowing for the solenoids to be slotted in place inside a compartment. Each solenoid is partitioned off from the others, allowing it to be held securely in place even if there are no other solenoids around it, so solenoids may be slotted into where ever they are needed and open strings can be played when needed. The top and bottom of each compartment is tapered to allow the solenoid to move without rubbing on the sides of the box reducing its movement. The legs attach to a flat bottomed oval nub on each side of the box body. The shaped nub prevents the legs from moving while attached, and acts as a guide to keep the legs straight. An M3 nut can be inserted into a slot in the back of the nub and a bolt inserted through each leg to tension and tighten them on to the box body, preventing the legs from widening across the bottom and from gripping the neck of the guitar. A slot is added on this which allows tension to be applied around the bottom of the leg. Fingertips may be printed to prevent the solenoids from damaging the guitar neck and fretboard as well as to provide a larger surface area to press down on the strings improving the accuracy of the device. These fingertips push over the end of the solenoids.

### 3.2 Picking module

The picking module (fig. 5) uses six arms, each with its own plectrum, one for each string. It consists of three components: mounting frame, mounting plate and strumming arms. There are currently two versions of the mounting plate: one to mount onto a single-coil pickup and the other to mount onto a humbucker. Both of these variations fit around the pickup below the strings to provide support for the rest of the picking module in both its height and positioning over the strings, while also preventing the strumming mechanism from moving around. To allow for different string heights, the guiding holes in the corners of the mounting plate can be cut down, reducing the gap between the strumming module and strings. The arms consist of 3D printed rods with built-in plectra, removing the need for any other parts being needed as well as making sure all the plectra are the same size and distance from the strings. The arms slot over the shaft of the servo motors while in the mounting frame. The mounting frame holds all the other components together, and is effectively a skeletonized box with slots for the strumming arms and servos. These slots make sure that the strumming arms are positioned correctly over the strings. The mounting frame also has holes for the servo motors. These holes are positioned on alternating sides to fit all of the picking arms into the needed space. This alternation of sides does not affect the picking sound. Strum commands are implemented in the software as rapid sequences of picks across the strings.



**Figure 4.** CAD model of the picking module

### 3.3 Muting module

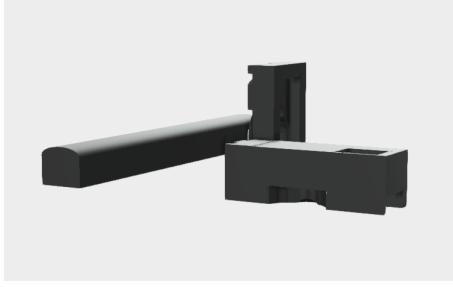
An optional palm-muting module may be attached to the picking module. It is formed of two CAD objects: a mounting bracket and the muting arm. The muting device mounting bracket attaches to the mounting plate of the strumming module and rests on the body of the guitar it main purpose is to hold the servo motor that controlled the muting this is done by lowering the muting arm over the strings, this muting arm is a bar that slots over the end of the servo allowing to be raised or lowered.

### 3.4 Serial port control protocol

An Arduino ([arduino.cc](http://arduino.cc)) controls the actuators via a serial port command protocol. Arduino C and Python serial calling code are included in the release. The protocol defines two types of 10-byte commands, one for the fretting module, and for both the picking and muting modules.

Fretting module commands such as S01F02UB00 (string 1, fret 2, up, no bend) begin with 'S' for string and a two (decimal) digit string number. The next three bytes are 'F' and the two-digit fret number. Currently only frets 1-3 are implemented but space is left for expansion up to 99 frets. The next byte indicates solenoid state (U)p or (D)own. The final bytes are 'B' and a two byte integer commanding the amount of string bending requested. (Bending is not currently implemented in the hardware.)

The picking and muting commands such as S01PV50E00 (pick string 1 at a velocity 50, now) begin with 'S' and a two-byte string number, usually ranging from 0-6 inclusive where 0 represents the muting device instead of the 6 physical strings. The next byte specifies the hit type such as (P)icking, (H)ammering or (T)remolo. (Currently only picking is implemented.) 'V' and two digits are then used for the velocity of the hit, 0-99. If the next byte is 'E' (end) then the action and any queued actions take place now. Otherwise the action is put in the queue. This enables multiple strings to be hit exactly simultaneously on an 'E' trigger if desired rather than in a sequence. Two final bytes are left blank for future expansions. If the mute is specified, the velocity is interpreted as a mute pressure, with 0 as no muting and 99 as full pressure. Currently only pressures 0 and 99 are implemented, i.e. the mute is either up or down – future work could implement the continuum.



**Figure 5.** CAD model of the muting module

#### 4. RESULTS

Included on the GitLab repository are videos showing the different features of the automated guitar in operation including (1) *House of the Rising Sun* chords and arpeggio playing; (2) multi-techniques demo with strumming, picking, hammer-ons and pull-offs; (3) *The Chain* riff showing picking multiple strings simultaneously (which a human player would find hard to do). A Python API and easy-to-use GUI are also provided, which wrap the serial protocol with higher-level commands such as to finger named chords and perform named strum and arpeggio types, and are used to program the demos.

The system is capable of playing any physically available combination of notes including some that would be impossible for a human player to reach. In the video demo, the fretting module only has 12 solenoids so is limited to play standard basic chords, though this is not an issue in general due to the modular design which means that additional frets can be covered by multiple fretting modules.

As the automated guitar must receive commands just before they need to be acted on, the serial interface needs to work with minimal latency. To achieve this a Baud rate of 9600 was used to record the demo videos. While this Baud rate is on the slower end of the standard rates it provides a transfer rate of 96 Kbit/s when using a 10 bit message, this transfer rate can be compared to the most commonly used communication music based protocols, the MIDI protocol has a transfer rate of 31.25 Kbit/s and allows for allow real-time communication between multiple devices.

It was found during testing that hammer-ons and pull-offs can be achieved using fretting module, if the solenoids were quickly powered on and off. The tone here is slightly different to and more metallic than a human player's, but provides a good starting point for further expansion, such as adding more human-like rubber fingertips to the solenoids.

The servos are specified to rotational speed of 60 degrees per 0.08 seconds, giving a maximum theoretical strumming speed of 18.8 picks per second due to each strum turning the plectrum by 40 degrees. However this is not achieved when running the system due to many factors such as the servo needing to swap direction. An empirical rate was measured to be around 10 picks per second. While this picking speed is fast for some types of music, such as western pop, and beyond what is needed by most human players, it can be considered slow for some types of music such as flamenco and when compared to expert guitar players who can achieve upwards of 20 picks per second with the world record being 33 per second. Higher speeds can be achieved by reducing the distance the plectrum moves, trading faster picking for lower velocity.

To calculate the maximum number of fretted notes that can be played per second, there are three main factors: the speed in which the strings can be pressed down in which to form the notes; the speed in which the automated guitar can strum the strings to play the note; and the speed at which the automation controller can receive instructions. These figures were gathered for the respective datasheets for the hardware components and calculated from the transfer rates of the software. The solenoid can actuate at around 10 meters a second, allowing it to travel a full stroke in 0.015 seconds although this will change depending on the voltage of its power supply. The servo is capable of picking each string 18.8 times a second. The serial protocol is capable of 4800 serial messages a second. These figures show that the servo is the bottleneck limiting the system to a maximum speed of playing 18 fretted note a second. However this will be lower real conditions at around 10 notes a second – this is still much quicker than a human player would be able to play and is not necessary for most songs.

#### 5. DISCUSSION

Automated instrument development and musical composition with them have been previously impeded by the lack of a standard, open, and cross-cultural platform. Research groups and composers have continually reinvented wheels, reducing time available for composition. We invite interested members of the computer music and OSH communities to use, standardize on, contribute to, improve, extend and fork our design to remedy this. The design is considered and intended to be modifiable not only to fit different types of Western guitar but also, with increasing effort, to be modified to fit related instruments from other cultures. An obvious next step could add a continuous pressure version of the mute. A module for bends – which would enable blues and some sitar-like styles – could next be implemented using similar technology but pressing down on the strings on the headstock above the nut, rather than attempting human-style horizontal bends on the fretboard.

#### 6. REFERENCES

- [1] S.-W. Leigh, A. Jain, and P. Maes, "Exploring Human-Machine Synergy and Interaction on a Robotic Instrument," in *Proceedings of the 2019 Conference on New Interfaces for Musical Expression (NIME)*.
- [2] S.-W. Leigh and P. Maes, "Guitar Machine: Robotic Fretting Augmentation for Hybrid Human-Machine Guitar Play," in *Proceedings of the 2018 Conference on New Interfaces for Musical Expression (NIME)*.
- [3] J. Lawrence, T. Howard, and S. Knueven, "Crazy J: a guitar playing machine," 2000. [Online]. Available: [http://ume.gatech.edu/mechatronics\\_lab/Projects/Fall00/group3/contents.htm](http://ume.gatech.edu/mechatronics_lab/Projects/Fall00/group3/contents.htm)
- [4] R. Vindriis and D. A. Carnegie, "StrumBot: An Overview of a Strumming Guitar Robot," in *Proceedings of the 2016 Conference on New Interfaces for Musical Expression (NIME)*.
- [5] J. Murphy, A. Kapur, and D. A. Carnegie, "Swivel: Analysis and systems overview of a new robotic guitar," in *Proceedings of the International Computer Music Conference (ICMC)*, 2013.
- [6] E. Singer, K. Larke, and D. Bianciardi, "LEMUR GuitarBot: MIDI robotic string instrument," in *Proceedings of the 2003 conference on New Interfaces for Musical Expression (NIME)*.
- [7] Gibson, "Meet Compressorhead — The World's Most Metal Band," 2012. [Online]. Available: <https://web.archive.org/web/20130416032139/http://www2.gibson.com/News-Lifestyle/Features/en-us/Meet-Compressorhead-The-Worlds-Most-Metal-Band.aspx>
- [8] Y. Suzuki, "Yuri Suzuki — Z Machines," 2013. [Online]. Available: <http://yurisuzuki.com/design-studio/z-machines>
- [9] F. Daniel K and G. Peter J, "Open-source hardware is a low-cost alternative for scientific instrumentation and research," *Modern Instrumentation*, vol. 1, no. 2, 2012.