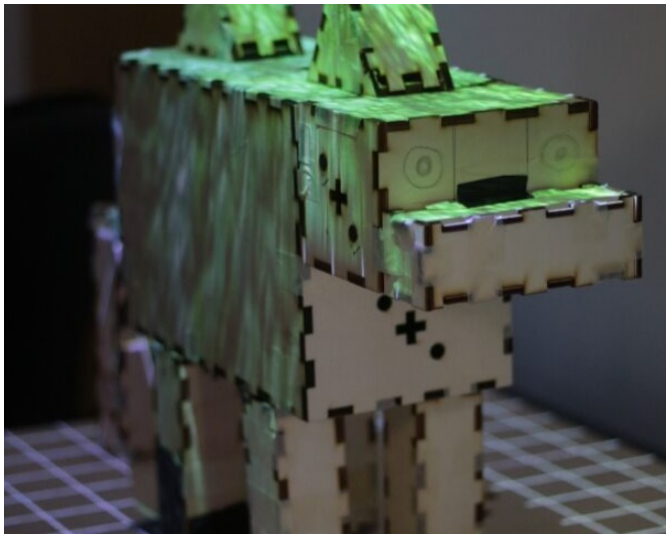


Magnosaur: An Interactive Dinosaur Puzzle

ALYSSA CHAN, ANGUS PURCELL, AZMIL ROSLAN, GEORGE WIGLEY, MAX PRASERTSAN, and JACK WALKER*

1 ABSTRACT

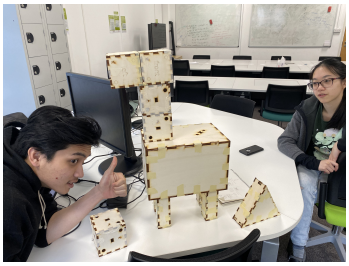
In this report, we detail the design, development, production, and evaluation of "Magnosaur", our tangible interface for a museum dinosaur exhibit. Most existing dinosaur exhibits use passive learning to convey information about dinosaurs; this project aims to make more engaging dinosaur exhibits by utilising active learning, which has shown to be more effective for promoting information retention than passive learning. Magnosaur is composed of a set of wooden boxes that act as the building blocks of a dinosaur. The boxes snap together magnetically and connect wirelessly. Upon connection, each block sends a signal to a central server, which digitally reconstructs the model on the table. Magnosaur projects the appearance and information about the closest discovered real-life dinosaur directly onto the model. We iterated for usability, enjoyment, and engagement with the device, as well as overcoming technical difficulties caused by the constraints and limitations of the project. We believe that Magnosaur is effective and addresses our research questions. We also surmise that our prototype would be markedly improved with a drastic increase to the time and resources available to us, taking into account that many of our applicable skills were learned during and for this project.

*All authors contributed equally to this research.

Authors' address: Alyssa Chan, fq20069@bristol.ac.uk; Angus Purcell, tt20799@bristol.ac.uk; Azmil Roslan, yw20551@bristol.ac.uk; George Wigley, zk20435@bristol.ac.uk; Max Prasertsan, zb20249@bristol.ac.uk; Jack Walker, ly18927@bristol.ac.uk.

2 INTRODUCTION

"Magnosaur" is a 3D interactive puzzle where users can build or solve a dinosaur silhouette with the given blocks. Once the silhouette is recognised by the system, facts about the recognised dinosaur silhouette will be displayed. According to a study by Bonwell and Eison (1991)[4], active exploration of a 3D object, rather than passively observing it, during initial learning could facilitate the recognition of the object. This could assist individuals with little to no knowledge about dinosaurs in learning more effectively and retaining the information displayed by "Magnosaur".



The interactive nature of "Magnosaur" promotes hands-on learning engagement, making it accessible as well as enjoyable for people of all ages and backgrounds, whether they are children fascinated by dinosaurs or adults looking to expand their knowledge.

While there are other interactive dinosaur exhibits around the UK, such as Yorkshire's Jurassic World at Yorkshire Museum,[2], these exhibits do not require users to manipulate 3D objects to receive information. Instead, users interact by watching videos on immersive displays.

2.1 Aims & Objectives

- Making the dinosaur exhibition more fun and interactive to learn from.
- Create a set of building blocks that can join up to make a dinosaur.
- Create a system to identify the connections between blocks.
- Create an algorithm that identifies the closest resembling dinosaurs from the building blocks.
- Project the related dinosaur on top of the finished blocks to represent their design.
- Make learning about dinosaurs more engaging and replayable.

3 RELATED WORKS

3.1 Active Learning

The term "*Active learning*" [6] refers to multiple learning models, which focus on the participant's responsibility to focus and learn [4]. The central concept is to focus on engagement, specifically in learning, discovering, processing and applying the information [17]. This concept is derived from the assumption that learning is an active mental process and that each individual learns in their own unique way.

Learning occurs through the accumulation of complex and personal experiences. [14] Active learning can be applied to museum exhibitions, such as an exhibition about dinosaurs. The aim is to draw the attention of the visitors and make that moment enjoyable to leave an impression on them.

By playing with these boxes, the visitors are doing *decision-making activities* [5]. This leads them to solve the puzzle by themselves, encouraging their brains to engage in solving the problem rather than passively receiving information

like the traditional exhibition would. Magnosaur is designed to allow the visitor to build a dinosaur from the blocks available. This will foster critical thinking skills [5] by introducing information analysis and evaluation and applying all these concepts to the physical blocks. The finished product is compared with our database of dinosaurs to match it with the closest dinosaur and then projected on top of the blocks.

3.2 Sony Block Jam

Block Jam is a Tangible User Interface that controls a dynamic poly-rhythmic sequencer using 26 blocks in a 2-D topological configuration. The blocks act as input devices for manipulating an interactive music system. Each block has a function and sound clip associated with it so that when the blocks connect with each other, the software on an attached computer recognises the organisation and plays the designed music[13].

Fig. 1. The design for Sony Block Jam from [12]

The intention behind Block Jam is to be physically organized and actuated and to attract the user's attention [12]. The idea of *'creating'* and *'discovering'* music in the Block Jam inspires the creation of Magnosaur. The target audience for the Block Jam design was adults, leading to a sophisticated interaction that promotes greater engagement. The device is shown in figure 1.

3.3 Topobo

Topobo is a 3D constructive assembly system with kinetic memory and the ability to record and playback physical motion. This input and output behaviour is unique among modelling systems. By snapping together a combination of Passive and Active components, users can quickly assemble dynamic biomorphic forms like animals and skeletons, animate those forms by pushing, pulling, and twisting them, and observe the system repeatedly play back these movements. Topobo can help people understand dynamic structures through interactive, tangible play[11]. We researched the possibility of using the dynamic connection method for the connection of each Magnosaur box.

3.4 SandScape

SandScape is a tangible interface for understanding landscapes and topography through simulations using sand. Users view these simulations through projections onto the surface of a sand model that represents the terrain. There are a

variety of different simulations that highlight different aspects such as height, slope, contour, shadows, or drainage[3].

Users change the landscape model by moving the sand and can see the resultant effects of computational analysis in real time as the projection updates on the sand. SandScape takes advantage of humans' natural ability to understand and manipulate physical forms while using simulation to help them understand a model representation.

The SandScape focuses mainly on improving graphical user interfaces (GUIs), virtual environments (VEs), and augmented reality (AR). The concept is to create an interactable interface that links directly to VEs. The sandbox offers the ability to manipulate sand and represent the topography on top of it, as shown in figure 2.

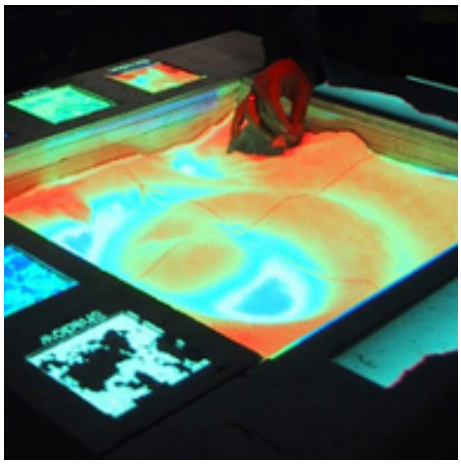


Fig. 2. SandScape exhibition from [3]

We are interested in the real-time image manipulation to be implemented in our dinosaur exhibition. The plan is to map an image on top of our blocks to show the dinosaur that closely resembles the final shape.

In conclusion, whilst Sony Block Jam creates an interactive learning environment to explore sound and music, it does not have a real time interface that updates in front of a user when connecting blocks. When building the blocks, a user must repeatedly check a computer to assess whether the outcome is desired. This is something that we have rectified with Magnosaur. As a user builds the blocks of the dinosaur, the skin projection updates in real time, letting the user know that a block as been updated successfully. Magnosaur also updates the projection to show a fact panel when a dinosaur has been successfully build. It is these considerations in the user interface that set Magnosaur apart from Sony Block Jam, and fills in the aspects that it is missing.

On the other hand, SandScape succeeds at real-time projection updates, giving the user a nuanced approach to interacting with an environment. However, this enjoyable experience is overshadowed by the fact that the uses of SandScape are limited. While it is an effective and interesting approach to learning about topology, it has not much practical use outside of that. Magnosaur can be adapted into many other scenarios outside of learning about dinosaurs,

from Minecraft to castles to architecture. The amount of ways this fundamental experience can be altered to create new learning opportunities sets Magnosaur apart, and should not be understated. Magnosaur is a versatile product that can provide a unique, enjoyable learning experience in many different fields.

4 DESIGN & PROTOTYPE

The main aim of the Magnosaur project is to create an interface that is engaging and memorable, in contrast to existing dinosaur exhibits. We intend for visitors to learn about dinosaurs on their own terms, promoting active learning to happen naturally by asking the user an open-ended task that requires critical thinking and reflection [15] rather than simple memorisation. We achieve this through making it a tangible interface that revolves around the user: information is projected directly onto the structure that the visitor creates as a direct consequence of their actions, making learning immersive, intentional, and entirely user-driven.

4.1 Final iteration

The plan for the Magnosaur changed throughout this project, resulting in the creation of various shaped boxes that represent each section of the body, splitting into head, body, limbs, tail/neck, and spikes. The magnets used are 10mm diameter neodymium magnets with 0.65 kg pull force. The connection between the blocks is still strong enough that the blocks do not fall by themselves. The Magnosaur blocks are light enough (about 200g or less) to prevent this from happening. A thin layer of glue was also placed on each magnet to cushion the magnets when the blocks magnetically snapped to prevent the magnets from breaking.

Each body part was made from a 6mm plywood sheet cut based on the blueprint provided in the Instructable and the measurement in table 1. The connection ports were then cut as cross-shaped holes into each side, following the pattern shown in figure 3. Holes were added to two of the corners, and subsequently filled with magnets. A Raspberry Pi Pico W [16] was fitted inside each box to act as a communicator between all the boxes. The final design for the connectivity between each box was an improvement from our two positive and two negative aluminium plates located at the edge of the cross-shaped hole. We altered the design to have three grounds, all connected in a U-shape via copper tape and a single positive general-purpose input/output or GPIO pin [7]. This was necessary due to time constraints, as this issue only became prevalent when we didn't have time to redesign the boxes. The benefit of this new setup is that once the boxes were connected, their grounds would be connected, while the GPIO pins were still also connected to the ground, causing a 'unified ground'. This was possible due to the fixed orientation of each connection port from the magnets, ensuring that all the connections will always at a 90-degree rotation, preventing the positive GPIO pins from coming into contact with each other. Unfortunately, this did add a slight reliability issue, where if the faces of the boxes were not perfectly flush, there could be a gap formed between one of the GPIO connections which would cause one of the boxes not to register the connection. We added a thin layer of folded paper under each connection terminal to add 'cushioning' to help alleviate this issue.

Fig. 3. Blueprint for the connection port.

Name	Width	Height	Depth
Head	150mm	100mm	100mm
Nose/Jaw	150mm	50mm	50mm
Body	150mm	200mm	300mm
Limb	50mm	50mm	100mm
Neck/Tail	50mm	50mm	200mm
Spike (from the base perspective)	50mm	50mm	70mm

Table 1. Measurement for the boxes in the Final iteration of Magnosaur.

4.2 First iteration

The first iteration of the Magnosaur was a set of $10x10x10cm^3$ blocks. These cube blocks are being used to test out other essential components of Magnosaur. First is the magnetic connection. This iteration used $10mm$ diameter neodymium magnets with $1.2kg$ pull force. However, the magnets are too strong, and some of them break when the block snaps. It also required a lot of force to pull the blocks apart. This is not ideal for when children are playing with Magnosaur.

For the microcontroller of the Magnosaur blocks, we opted for the Raspberry Pi Pico. The main reason is that after testing, Arduino Nano tends to get damaged more easily compared to a Pico when there are magnets nearby. Furthermore, Pico supports MicroPython which is our preferred programming language for its ease of use. Using the regular Pico required a physical connection between the blocks to transmit data to each other. Initially, we planned on using pogo pins, however after testing we discovered that an inability to slightly twist boxes as you pulled them apart, in addition to the cost and fragility of a pogo pin, made them a poor choice for this project.

4.3 Second iteration

For the second iteration, we changed the design of each box. The feedback provided by early testers - as shown in 5 - suggested that we should make the boxes bigger for ease of use, quicker build times and satisfaction factor when clicking boxes together. In this iteration, the boxes were separated into 5 different categories: Head, Body, Limb, Neck/tail, and

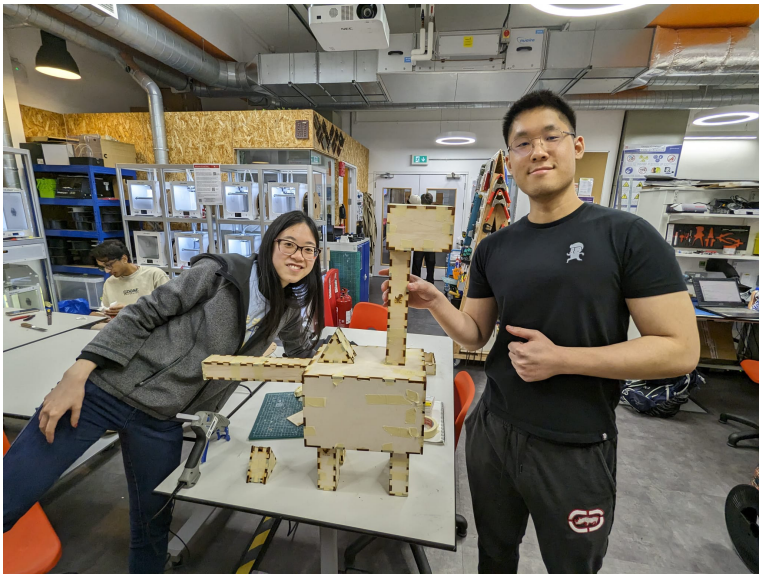


Fig. 4. The final iteration of Magnosaur with Alyssa (Left) & Max (Right) for scale.

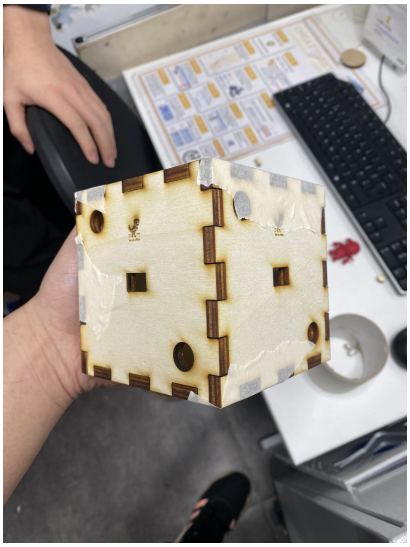


Fig. 5. The first iteration of the magnosaur building block.

Spike. Each has its own dimension, as shown in the table 2.

We designed each piece to fit within a voxel grid, with each voxel being a 50mm x 50mm x 50mm in three-dimensional space. A voxel is akin to a pixel except in 3D space; we use a voxel grid to hold the digital representation of the wooden

Name	Width	Height	Depth
Head	100mm	100mm	150mm
Body	200mm	200mm	300mm
Limb	50mm	50mm	150mm
Neck/Tail	50mm	50mm	300mm
Spike (from the base perspective)	60mm	60mm	52mm

Table 2. Table containing the measurement of iteration II boxes.

structure created by the user, which is in turn used to calculate the projection mappings.

We also upgraded the regular Raspberry Pi Pico to the Raspberry Pi Pico W, a Pico with WiFi capabilities. This improvement solved several issues. Firstly it removed the need for an analogue connection between the Picos, as they could wirelessly communicate their data to a centralised server using WiFi. This allowed us to use small buttons to indicate when a box's side formed a connection with another box, as well as fix the problem of limited analogue pins present on a Pico. The buttons introduced instability to the system because they were prone to error. We deemed the buttons infeasible and moved on to the third design.

4.4 Third iteration

In the third iteration of the device, we improved the connections between boxes. Since GPIO pins limited us to switch-like connections, the software knew which two faces of the boxes were connected together using a very short timer. The main issue with this is that someone could press a button on another box, or a different face on the same block, and the software would become confused as to which two faces were actually connected. To solve this we decided to use conductive terminals on each face. This would allow the boxes to retain the button-like connection mechanism while making it nearly impossible for an accidental press to occur.

Because of the nature of our magnetic connection, there is only one orientation a block can connect to another block. This means that with the same magnet setup on another face, the terminals are effectively flipped and then rotated 90 degrees. The rotational nature of these connections meant that for a consistent connection, we would need evenly spaced terminals in multiples of four. Thus we decided on a cross-shaped terminal with two GPIO pin connections at the vertical sides of the cross, and two ground connections at the horizontal sides of the cross. This layout allowed for the grounds and GPIO pins to always connect between two blocks, and added consistency if the block was not perfectly flush.

To make the terminal connections on each face, we initially used aluminium foil, as it was easily accessible and a good conductor. Its flexible nature also meant that it could be easily manipulated into the necessary shapes. We then had wires directly from one GPIO pin to the two foil output terminals, and two ground wires going to the foil ground terminals. While this did work in practice, aluminium foil proved to be too fragile for constant use and would often break or rip during testing.

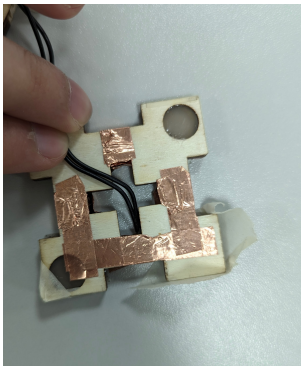


Fig. 6. Caption

4.5 Forth iteration

After encountering the problem with the fragility of the aluminium foil, we turned our attention toward other conductive options. The solution we found was copper tape. The tape was much thinner than the foil we previously used, and so the resistance ended up being a lot higher. There was also an issue of it being so thin that if the box faces weren't perfectly flat, sometimes terminals wouldn't connect. To work around these issues, we fed the wires from the Picos directly onto the outside face, and simply covered it with the tape as a means to increase surface area. We also placed some folded paper under each connection to add some 'cushioning' to allow the terminals to be slightly raised, but compress to allow the blocks be flush if needed. At this stage, we continued to use the 2-GPIO, 2-Ground setup that we did in the previous iteration.

We planned for our end product to be powered by battery, rather than plugged into power. Unfortunately, we didn't get access to battery holders and batteries until very close to the end of the project. Once we began testing with battery power, we discovered an issue where the Picos must have a 'common ground', or else they will not detect if they are being pulled down. At this point, we could not alter the design of the boxes, as making new ones would take longer than the time we had. This led to us altering our design by removing one of the GPIO pin connections on the cross and connecting the three non-GPIO terminals in a U shape. This solution meant that when the connection between blocks occurs, the grounds of each Pico would be connected to each other, creating a 'common ground', and the GPIO terminals would also connect to this unified ground.

4.6 Software

4.6.1 Server.

Initially, our approach was to construct a tree data structure using a node system where each box is a node, with a recursive check on the types of nodes to see whether they are identical to a known tree that represents a dinosaur. However, after evaluating the implementation of the magnetic boxes, we determined a more intuitive and efficient method. We opted to represent the boxes in 3D space, where each box's placement and dimensions are tied to the type and position of the preceding box. This approach also makes the projection of dinosaur images onto the boxes easier as there will be a direct tie into the physical space. By adding the boxes sequentially, greater flexibility is afforded by

allowing many different shapes and configurations and, therefore, different types of dinosaurs.

Each box is treated as a separate object of the class BoxNode, and since the number of boxes doesn't change, each box can be instantiated during the setup of the code. The class BoxNode defines a box's location by the locations of its two opposite corners, one corner is always the corner with the smallest (x, y, z) coordinate and the other has the largest. The class also contains the orientation of the box, which is defined by which of the box's faces are facing down (towards a smaller y value). This means that all boxes start with an orientation of "bottom", and this is changed as the boxes are rotated. The class also contains a box's type, so its size can be found, and an ID number, so boxes of the same type can be differentiated.

The first box that is added to the 3D array that represents the boxes in space, is the root leg at position (7, 0, 4). This leg will not change position and will always be present in real space. This leg serves as a position to build the rest of the dinosaur.

When the software gets a message from two Pico Ws saying that they have both been pressed, then it infers that these two have been connected to each other as the time between presses is very small and they can only make connections with another dinosaur connection, meaning that connections have to come in pairs. An array of existing boxes is checked to see which of these boxes are already represented in the array, and which box isn't. After identifying which box is already connected, the orientation and connected face of that box is checked and compared against the face of the new box that has received a connection, and the new box is rotated into the orientation so that the two connected faces can align. The new box is then translated so the specific locations of the connections on each box touch each other. After this, the orientation of the new box is updated to which face is facing down. The existing box will never move during this process, it is purely aligning the new box to fit the existing one.

The function that rotates a box into the correct orientation is shown below. The rotation matrix for rotating the location vector (which is the array that holds locations of opposing corners of a specific box) counterclockwise around an axis by a specified angle is given by [8]:

$$R = \begin{bmatrix} a^2 + b^2 - c^2 - d^2 & 2(bc + ad) & 2(bd - ac) \\ 2(bc - ad) & a^2 + c^2 - b^2 - d^2 & 2(cd + ab) \\ 2(bd + ac) & 2(cd - ab) & a^2 + d^2 - b^2 - c^2 \end{bmatrix} \tag{1}$$

where:

angle = angle in radians,

axis = axis of rotation in the form of a 3D array,

bc, ad, ac, ab, bd, cd = $b * c, a * d, a * c, a * b, b * d, c * d$,

$a = \cos\left(\frac{\text{angle}}{2}\right)$,

$b, c, d = -\text{axis} \times \sin\left(\frac{\text{angle}}{2}\right)$,

and $\times$ denotes the cross product.

To apply this rotation to a vector $\mathbf{v}$, the following transformation is performed:

$$\mathbf{v}_{\text{rotated}} = R \cdot \mathbf{v}^T \tag{2}$$

where $\mathbf{v}$ is the original location vector and $\mathbf{v}_{\text{rotated}}$ is the rotated vector. After the matrix multiplication, transpose the result to get the final rotated vector:

$$\mathbf{v}_{\text{rotated}} = (R \cdot \mathbf{v}^T)^T \tag{3}$$

In Python, this transformation is implemented as:

```
self.location = np.dot(rotation_matrix, self.location.T).T
```

where np.dot is the function for matrix multiplication, rotation_matrix is the rotation matrix, and self.location is the original vector. np.dot is a function from the numpy library [10]
These functions are arranged so that a box can be rotated counterclockwise around the z-axis by 90 degrees by calling:

```
self.rotate([0, 0, 1], 90)
```

When the location and orientation of a box have been positioned correctly, a function is called that takes the location vector of the box and writes a `1` at every voxel inside the box. This ensures that the 3D array will contain a `1` at every location where a box is present and a `0` otherwise.

The final step of this process updates the projection with the new box every time a box is added by sending the updated 3D array to the projection function.

This whole process is repeated every time two Pico Ws register a connection at the same time. By doing so we can update the project with accurate information every time a new box is connected to the structure.

Removing a box when two Pico Ws register a disconnect takes fewer steps. When the initial disconnect is registered, the two Picos' IDs are checked against a list of connected Picos in the order that they connected, with no repeats, to see

which one was connected last. The Pico that was connected last out of the two has to be the one that is removed. The Pico's corresponding box object is then ascertained through its type and ID. Using this box object, a copy of the function that writes 1 at voxel locations is used but it writes `0` instead. This removes the box from the 3D representation array. After this, the box is removed from the list that holds connected boxes, its first corner's position is set back to (0,0,0), and the orientation is changed back to "bottom".
This process essentially reverts the box object to its state when first instantiated before any connections were registered.

4.6.2 Pico Ws.

The software on the Pico Ws consists of two main parts: the interaction function and the WiFi connection function. The first task we had was to connect the Pico Ws to the central server. To do this, we used Microythonʼs network module and created a function called 'connect_to_internet'. This function takes a Service Set Identifier (SSID) and a password and attempts to connect to the WiFi with those parameters. While the connection is pending, the onboard LED flashes will add a visual aid for the user to know if the device has yet to connect. Once a certain period of time has passed, set in the variable 'max_wait', without a connection, the device will hard reset itself in order to attempt the process again. This was done because there was an inconsistency with the Picos being able to reconnect to a WiFi point they had been connected to previously.

The second part of the Pico W software is interaction handling. Most of this is encompassed by the 'send_interaction' function. Here, we make use of Micropythonʼs socket module to allow Pico to send and receive data from the server. The function takes an 'interaction', a 'face', and a 'retry_limit'. The retry limit is an optional integer and usually won't need to be altered. The 'interaction' and 'face' are both strings. The most commonly sent interaction for each Pico will likely be the heartbeat. We set up a timed 'heartbeat' message that will send the interaction 'xHEARTBEAT', with the face set to 'h', to the server, and we expect a response. If a response is received, then the code will continue, else it will try to send this message again up to the retry limit. At this point, it will assume that the WiFi it was connected to has been turned off or the connection is otherwise not working, and the Pico will reset. This was necessary because there was no way for the Pico to detect if the WiFi connection it was using had been disconnected outside of trying to interact directly with it. The other interactions the client sends are 'xPRESS' and 'xRELEASE'. These interactions are used to tell the server when a face has a connection and have been set to only message the server once on press and once on release. For the face variable, will be set based on where the GPIO pin is connected. 'face' follows the structure 'top.1' where 'top' is whichever face the pin is connected to (top, bottom, front, back, left, or right) and the '1' is replaced by whichever number of pin is connected for that face. This number starts from 1 and counts from top left to bottom right, assuming you are viewing the box from the side, where in a typical full setup the head is facing your right. For a lot of boxes, this number is superfluous as there are only singular connections per face, however for the larger blocks, such as the body, it is necessary.

There are a few global variables in the code for the Pico Ws that need to be set based on the users. Each pico will have a name. This needs to be set manually for each Pico and follows the format 'l/1' where 'l' is replaced with the first letter of whichever part type it is (leg, body, head, tail, etc) and the '1' is simply a unique integer for that piece. The server IP address and port will also need to be set. This is simply the IP address and port used in the sever code. There is also a heartbeat timeout variable that can be altered which is the gap you want between heartbeat messages in seconds.

4.6.3 *Projection.*

To convincingly project onto real-world 3D geometry, we evaluated several options. Initially, we intended to calibrate a projector and camera pair and then use that calibration information to compute a rasterization of the geometry onto the image plane of the projector. This can be thought of as treating the projector like a camera in which geometry is "seen". This would allow us to know the position of each block in the screen space of the camera defined by a polygon of (x, y) pixel coordinates. Whilst this approach likely would have worked, it would have required substantially more work than required for our scenes, and thus, we opted for another method.

This method involves treating the projector like a point light in the context of computer graphics. We can then compute a shadow map from the perspective of the projector. This involves determining the regions of the table (we will assume this to be an infinite planar surface) that are occluded by the geometry. If we ensure that the projector's image aligns with the table during setup, we can create a mapping from world space coordinates on the plane to pixel coordinates in the projector. In other words, if we define the top left corner of the table to be at the origin of the system $(0, 0, 0)$ and the bottom right to be at $(tablewidth, 0, tabledepth)$ it follows that the projector's image will map from $(0, 0)$ at the top left to $(projectorresolutionwidth, projectorresolutionheight)$ at the bottom right of the table. This setup is illustrated in 7

Once we have computed the "shadows" cast by the geometry, we will have a polygon in the world space of the form $x, 0, z$, allowing us to linearly map from world space to screen space. To perform this projection, we can define a ray of the form 4

$$P(t) = Origin + t(Direction) \tag{4}$$

where $P(t)$ defines the point on the ray. $Origin$ defines the origin of the point as a vector. $Direction$ defines the direction of the ray as a vector. t defines the distance to travel in $Direction$ from $Origin$.

To find the projection of a point to the plane, we can solve for a value of t where $y = 0$. More precisely, we can define the ray equation in one dimension, which in our use case is y giving 5

$$final_y = Origin_y + t(Direction_y) \tag{5}$$

Where $final_y$ is a scalar defining the y value along our ray. $Origin_y$ is a scalar defining our starting y. t is a scalar defining how far to move along the ray. $Direction_y$ is a scalar defining the direction of the ray and the distance between the projector and the point.

By setting $final_y = 0$ and rearranging to solve for t we get 6

$$t = \frac{-Origin_y}{Direction_y} \tag{6}$$

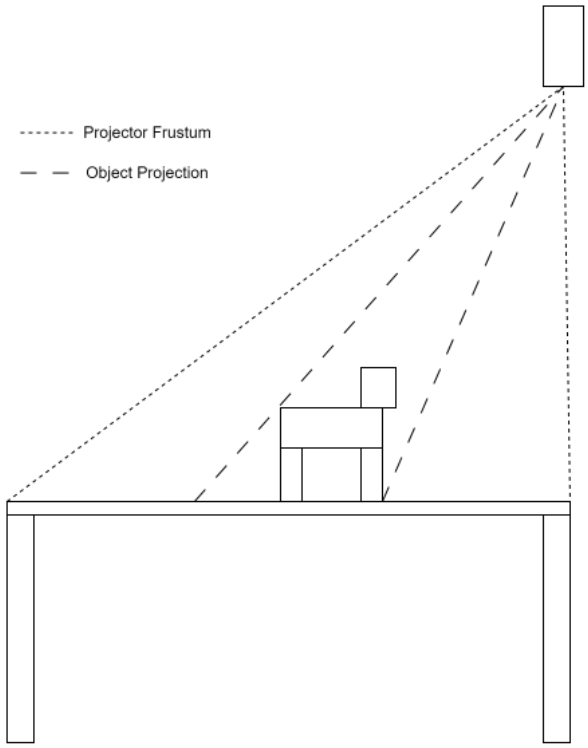


Fig. 7. A diagram illustrating the projection setup, the projector is shown above the table, and the lines illustrate the different projections occurring.

We can then substitute t into 4 and find the point of intersection with the plane. For each block, we can repeat this process for every vertex and draw a convex polygon from them into our mask, indicating a region of occlusion. For all of the above equations 4 5 6, the vectors should not be normalised as they represent world space translations.

When we send a frame to the projector, we have 2 textures for the table projection and the dinosaur projection using the mask to blend between them. Our current implementation uses the same texture for every block, but a future implementation could track which block produced which polygon and then perform projective texture mapping. This would allow for specific textures on specific regions of each block. Adding this feature would require the creation of a full rendering pipeline with support for texture coordinates, projective transformations and depth testing which was out of scope for this project.

The results of the projection system can be seen with a block in 8 and without a block in 9. The full projection code has been included in the project files and is designed to be reusable and modifiable for other use cases.

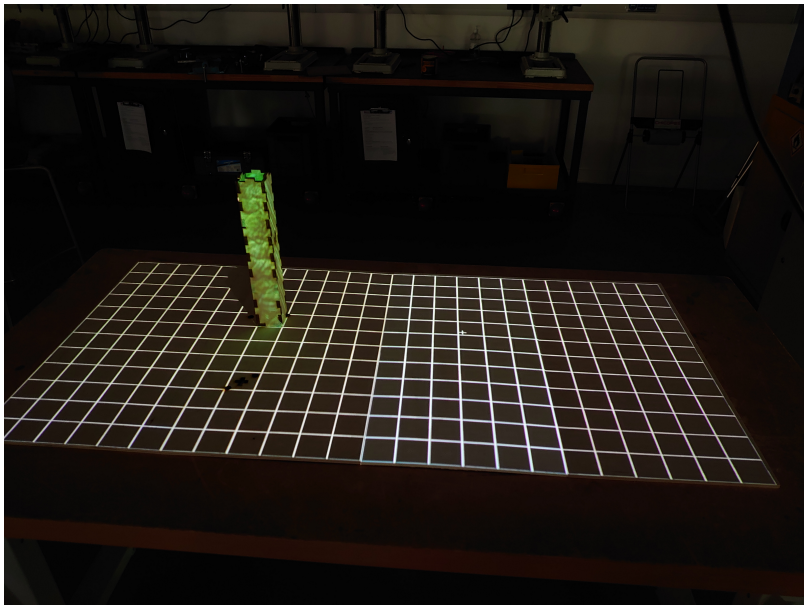


Fig. 8. An example of a 1x6x1 block demonstrating the projection system. The texture is projected onto the block whilst the remainder of the table is a grid.

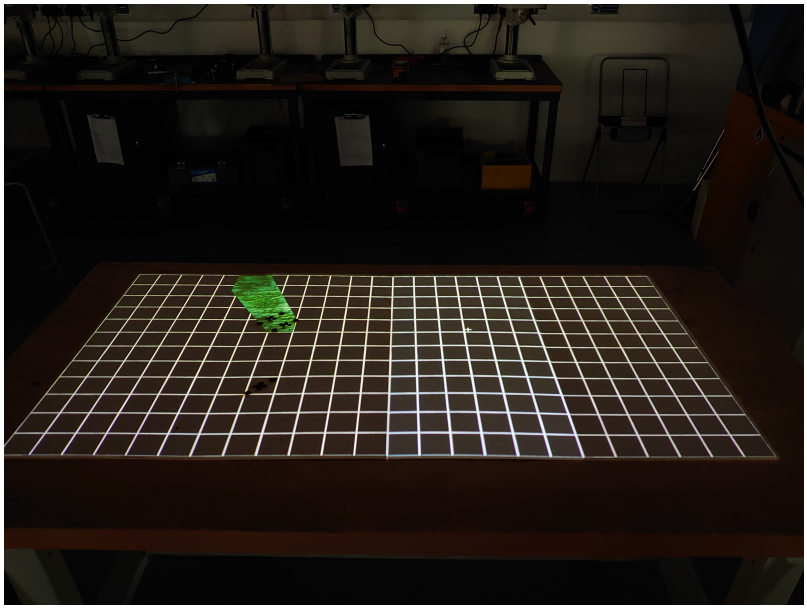


Fig. 9. An example of the projection with no block included showing the "shadow" that the block would create.

5 HUMAN STUDY

During the initial stages of the development cycle, we created a survey to investigate the needs and opinions of museum visitors. 10.

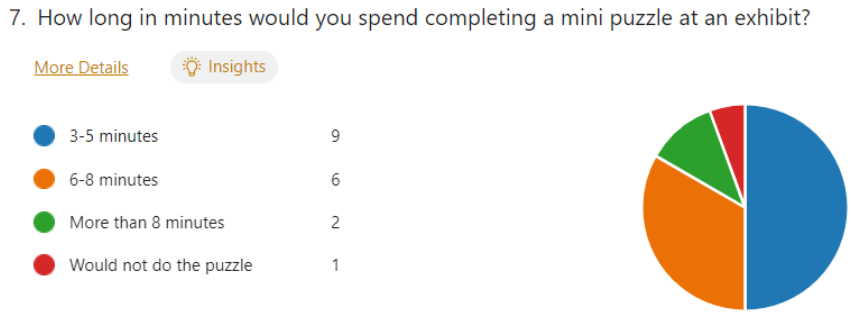


Fig. 10. The graph represents the participant's willingness to solve a puzzle.

Our qualitative results were processed using thematic analysis, an umbrella term that is often used to refer to reflexive thematic analysis, an approach that was pioneered by Virginia Braun and Victoria Clarke.[1] It follows a six-phase process:

(1) Familiarisation: reading and re-reading the data to become immersed and intimately familiar with its content, and making notes on initial findings for each datum but also in relation to the dataset as a whole.
(2) Coding: not to be confused with programming, coding is generating succinct labels that capture and evoke important features of the data that might be relevant to addressing the research question. There are multiple rounds of coding, the results of which are subsequently collated for later phases.
(3) Generating themes: examining the codes and collated data to identify patterns of meaning across the dataset. "Themes" - central concepts or ideas underpinning these patterns - are proposed, and data is collated by candidate themes.
(4) Reviewing themes: the proposed themes are checked against the collated data and the dataset as a whole, and evaluated. Do they satisfactorily explain the data? Themes can be further developed, being split, combined, or discarded.
(5) Defining and naming themes: a detailed analysis is developed for each theme, defining the scope, focus, and meaning of each theme. Each theme is also given an informative name.
(6) Writing up: weaving together the analytic narrative and data extracts and contextualising the analysis in relation to existing literature.

One theme we found in the survey was that people found existing dinosaur exhibits "not interactive enough", some even specifically stating "too much reading" or "repetitive". Hence, we decided to come up with a tangible interface to promote a feeling of engagement with the device.

One concern with early iterations was that the connections between the blocks felt "mushy", and parties were unsure if the blocks were securely and correctly connected. To rectify the former concern, we tested different configurations

of magnets: we found that the "snappiest" and strongest connections were achieved by affixing the magnets as flush with the outside surface as possible, with nothing but a layer of tape over each to prevent them from being pulled out. For the latter concern, we increased the thicknesses of the terminals to maximise the chance that a correct electrical connection would be established first try.

As Figure 7 illustrates, 50% of participants were willing to spend 3-5 minutes completing the task. As a result, we decided to move away from using a large number of small cubes, exchanging them for a series of larger cuboids and prisms of different shapes and sizes. Though 1x1 boxes were more granular and provided greater freedom with the build, it would've taken up to one hundred and eleven (111) 1x1 boxes to create structures equivalent to those produced with the final iteration. Using cuboids of various lengths instead drastically cut down on both the time the interaction required and the resources required to produce it. Further tests had no complaints of tedium, complexity, or boredom. Additionally, using larger blocks aligned with the "scale" theme we found in the data - people like it when the dinosaur exhibits are big.

6 CRITICAL EVALUATION

There are a few issues with the current design. One of the main problems that we faced was that the boxes were made out of plywood, which brought about the issue of warping and splinters. We took measures against this by sanding the edge of each box to "break" [9] the edge. With better resources and more time, the use of 3D printers can be considered to avoid this problem and to make the blocks lighter.

Other than that, the blocks had inconsistent physical connections with each other. When we were testing and deciding on the strength of the magnets that should be used, each tester block only had one Pico W inside without batteries. 0.65 kg pull strength worked fine with the tester blocks. However, when the first complete Magnosaur prototype was ready, we noticed that the weight of the two AA batteries caused the blocks not to attach firmly and fall on some occasions. Due to a restricted budget, we were unable to order a new set of much stronger magnets. So, a stronger magnet is highly recommended for any future iterations.

The next issue we faced with the physical design of the box was inconsistency with the connection of the custom terminals. Initially, the custom terminals were made out of an aluminium tin foil. However, tin foils do not stick, and tape is required. Tapes are not conductive, and if the terminals are not made carefully, the tape could cover the conductive bit of the terminal. Small bits of wires are also required to bridge the 3 ground terminals. Hence, we decided to use copper tape. Copper tape is conductive on both its adhesive and non-adhesive surface. Copper is also more conductive than aluminium. [18] This made it easier to connect the wires to the terminals and to bridge the 3 ground terminals. Initially, the wires that connect the terminals to the Pico Ws were connected at the back of each port. However, we discovered that the terminals are unable to send signals to the Pico Ws. So, the wire was taped on the surface of the terminals. This adjustment improved two things: the signals can now be detected by the Pico Ws, and the protruding wires ensure a stable connection between terminals as the magnets are able to push the two terminals together.

However, the connection between each box could have been implemented better with more resources, both material and time. The current connections rely solely on thin strips of copper tape on top of exposed wires. Copper tapes are fragile and easily break when used for prolonged periods. The prototype managed to show connectivity between

multiple boxes, but it did not pass the stress testing, which included dropping the boxes, aggressively slamming the connection together, and sliding the boxes off to remove the connection.

Following a better implementation of the box connections, another improvement would be to improve the method of data transmission. Currently, all the Picos connect through WiFi to a nearby server and have a binary on/off system to detect connections. While this worked fine in practice, it was also reliant on the wireless connection being very fast and two blocks being fully connected before another block was added to the structure in any way. If we have a microcontroller with a large number of analogue pins, we could allow them to communicate with each other directly when blocks connect rather than indirectly through a server.

This then gives two alternative and more robust options for block connection. The first option would be to have a similar implementation to the current system; however, instead of a block just detecting a connection on a specific pin, it would also be able to send and receive the ID of the block it is being connected to. This would allow multiple blocks to be connected at any time, even simultaneously, and there would be no issue of the server being confused about block locations. The second potential solution with these microcontrollers would be to have the root node wired to the server. From here, every block that connects could send its ID and connection pin to the block it attaches to. The block it attached to would then add that to the corresponding ID and connection pin for itself and send this data in a chain through all the blocks to the root node, which is in turn connected to the server to perform the 3D calculations. This second method would likely require a lot more work but would completely eliminate the need for a wireless connection.

In the current iteration, the blocks are using AA batteries to power the Pico Ws and are not rechargeable. The blocks must be disassembled to swap the batteries for the Picos. For future iterations, rechargeable lithium-ion batteries can be used. This eliminates the need to disassemble each block when they run out of battery. This can also allow the implementation of wireless charging, which minimises the need for any open ports on the blocks. This is more environmentally friendly considering that this project could be expanded by using smaller, granular blocks, i.e. 1x1 voxel blocks, instead of the current iteration's large blocks. More granular blocks would allow users to be more creative and build more types of dinosaurs.

7 CONCLUSION

Magnosaur is a hands-on, tangible interface with immersive elements; as far as we are aware, it is the first entirely user-driven dinosaur museum device of its kind. We believe it successfully encourages active learning and active engagement with the content it aims to teach. Additionally, user tests indicated it was more fun and re-visitable than traditional dinosaur exhibits. Despite all of the technical challenges imposed by the limitations of our project, we believe that Magnosaur successfully addressed our research question.

Provided we had more time resources at our disposal, the next iteration of our prototype would have rechargeable batteries, and microcontrollers with larger quantity of analogue pins, to improve the method of block connection. We would also connect the root node directly to the server to boost the robustness of the connections on the software side; the terminals would be manufactured out of a sturdier material to promote more secure connections on the hardware side. We would also 3D-print the components, in order to improve the structure's weight and mitigate issues with

splinters and warping.

However, in its current state, Magnosaur can be used as a framework to create nuanced learning experiences for a variety of different topics. The idea of having constructable 3D objects which can be projected onto can be used in a multitude of ways to incentivise learning. From using the method to create an interactive Minecraft exhibit, or to help teach architecture, or to create a buildable castle exhibit. Magnosaur represents a revolutionary new way to improve interactive learning.

REFERENCES

[1] [n. d.]. Understanding TA | Thematic Analysis. https://www.thematicanalysis.net/understanding-ta/. (Accessed on 04/05/2024).

[2] 2024. Yorkshire Museum. Museum Gardens, Museum Street, York, YO1 7FR. https://www.yorkshiremuseum.org.uk/exhibition/yorkshires-jurassic-world/.

[3] Ben Piper Carlo Ratti Professor Hiroshi Ishii ao Wang, Assaf Biderman. [n. d.]. SandScape. *Tangible Media Group* ([n. d.]).

[4] Charles Bonwell and James Eison. 1991. "Active Learning: Creating Excitement in the Classroom". *ASHE ERIC Higher Education Report No. 1* (01 1991).

[5] Cynthia Brame. 2016. Active learning. *Vanderbilt University Center for Teaching* (2016).

[6] Isabelle D Cherney. 2011. Active learning. (2011).

[7] Warren Gay. 2014. *Raspberry Pi Hardware Reference.* https://doi.org/10.1007/978-1-4842-0799-4

[8] Gene H Golub and Charles F Van Loan. 2013. *Matrix Computations.* JHU Press.

[9] Garrett Hack. 2011. How to Break an Edge. *FineWoodWorking* (2011).

[10] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature* 585, 7825 (Sept. 2020), 357–362. https://doi.org/10.1038/s41586-020-2649-2

[11] Professor Hiroshi Ishii Hayes Raffle, Amanda Parkes. [n. d.]. Topobo. *Tangible Media Group* ([n. d.]).

[12] Hiroaki Nakano Henry Newton-Dunn and James Gibson. 2003. Block Jam: A Tangible Interface for Interactive Music. *Journal of New Music Research* 32, 4 (2003), 383–393. https://doi.org/10.1076/jnmr.32.4.383.18852 arXiv:https://www.tandfonline.com/doi/pdf/10.1076/jnmr.32.4.383.18852

[13] Martin Kaltenbrunner. 2002. Sony Block Jam. (2002).

[14] David Kolb. 1984. *Experiential Learning: Experience As The Source Of Learning And Development.* Vol. 1.

[15] Active Learning. 2013. Strategies to promote critical thinking and active learning. *Teaching in Nursing E-Book: A guide for faculty* (2013), 258.

[16] Raspberry Pi Ltd. [n. d.]. Raspberry Pi Pico and Pico W. ([n. d.]).

[17] Richard Mayer. 2004. Should There Be a Three-Strikes Rule Against Pure Discovery Learning? *The American psychologist* 59 (01 2004), 14–9. https://doi.org/10.1037/0003-066X.59.1.14

[18] Tai Sin Singapore. [n. d.]. Electrical Conductivity | Copper vs Aluminum Cable. ([n. d.]). https://www.taisin.com.sg/electrical-conductivity/ Accessed: 4th May 2024.

8 APPENDIX

A VIDEO LINK

https://www.youtube.com/watch?v=wf-kKVuHDaQ

B INSTRUCTABLE

https://www.instructables.com/Magnosaur-Interactive-Dinosaur-Puzzle/

C CONSTRUCTION PROGRESS

Received 9 May 2024; revised 12 March 2024; accepted 9 May 2024

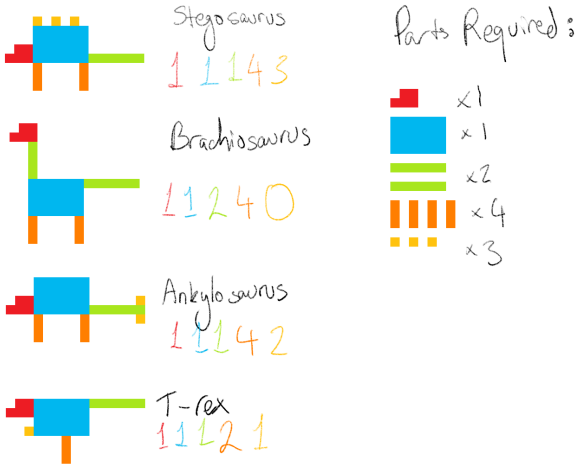


Fig. 11. Concept sketch

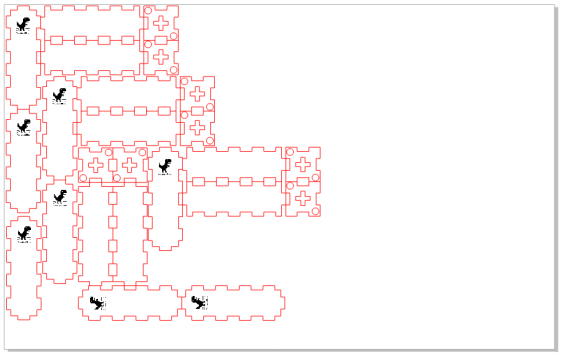


Fig. 12. Laser cutting pattern for limbs.