MeshChain: Secure 3D model and intellectual property management powered by blockchain technology

Hunmin Park[1], Yuchi Huo[2], and Sung-Eui Yoon[3]

[1]KAIST, mathematicianscott@gmail.com (95phm@kaist.ac.kr),
[2]KAIST, huo.yuchi.sc@gmail.com,
[3]KAIST, sungeui@gmail.com

Abstract. The intellectual value of digitized 3D properties in scientific, artistic, historical, and entertaining domains is increasing. However, there has been less attention on designing an immutable, secure database for their management. We propose a secure 3D property management platform powered by blockchain and decentralized storage. The platform connects various 3D modeling tools to a decentralized network-based database constructed on blockchain and decentralized storage technologies and provides the commit and checkout of the 3D model to that network. This structure provides 3D data protection from damages and attacks, intellectual property (IP) management, and data source authentication. We analyze its performance and show its applications to cooperative 3D modeling and IP management.

Acknowledgements

This work was supported in part by NRF (2019R1A2C3002833) and Starlab (IITP-2015-0-00199).

1 Introduction

3D models have become one kind of valuable property in many domains. For example, many scientific studies are targeting the study of 3D structures of microorganisms. Modern artists materialize their thoughts as 3D model artworks. Designers claim authorities to 3D models as the products of their intellectual endeavors. The scanning of relics and celebrities bestows 3D models' historical values. In the domain of computer graphics, the 3D model is one of the primary inputs and outputs for different subjects, such as rendering, geometry, and animation [17,34,18,2,36].

While 3D models have become nonnegligible intellectual properties of society, organizations, and individual persons, there has been less attention on designing immutable secure mechanisms to preserve, manage, cooperatively produce, and authenticate 3D models.

Beyond the actual 3D data, the intellectual rights of the authors are even more vulnerable. Nowadays, the internet is the most popular repository to disseminate digital data, including 3D models. While people spread and share 3D models through web pages or social networks, the data is vulnerable to tampering and plagiarizing. A reliable platform to authenticate the 3D data and track down the history of the data will help the creators protect their intellectual properties.

The process of creating 3D models also requires data security and intellectual rights management, but those issues have rarely been studied. Nowadays, the 3D models used in movies and games are becoming complex, which sometimes overwhelms individual designers. So collaborative modeling has gained increased attention and became one of the standard workflows. Currently, the existing 3D model platforms are local or based on centralized networks. For example, some collaborative 3D modeling platforms [1,26] use centralized networks that depend on the central point (central node or central group of nodes) to connect the 3D modeling tools. In such a network, all data in a network should pass through the central point.

Building a 3D management platform on a classic centralized network enjoys the advantage of simple deployment. However, platforms on centralized networks have their limitations. First, maintaining the central nodes is expensive and not suitable for flexible collaborations within communities. Second, the network and data are vulnerable to network attacks or physical disasters. Third, there is no strong protection mechanism to prevent the tampering of integrity, security, and property of the intellectual asset from the central nodes.

Our contribution in this paper is the proposal of a decentralized 3D property management platform. The decentralized platform, powered by blockchain and decentralized storage techniques, connects various 3D modeling tools to local clients. Specifically, we adopt the blockchain technique for managing intellectual properties and the decentralized storage technique for storing the (large) 3D data safely. Such design gives the following benefits to 3D property management:

- Flexibility enabling various applications of the platform.
- Decentralized 3D data, transparent intellectual property (IP) management, and IP protection.
- Invulnerability to network failures and physical disasters.

In this paper, we also show the implementation of our method built on Ethereum [6] and Swarm [10]. We analyze its performance overhead and show its applications to 3D collaborative modeling.

2 Related work

In this section, we discuss decentralized data security, cooperative 3D modeling, and decentralized version control that are related to our work.

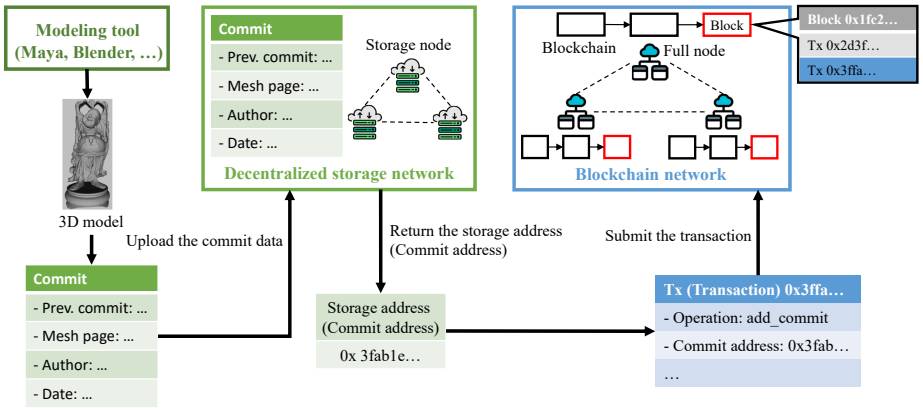

Fig. 1. The overall structure of the proposed platform and the fundamental operations. The platform contains three parts: the local client, index blockchain network, and decentralized mesh network. Nodes within the index blockchain network and decentralized storage network connect through peer-to-peer connections (dotted lines).

2.1 Decentralized data security

One common method of protecting crucial digital data is decentralizing the data. Blockchain is one of the hottest decentralized techniques in recent years [32,33,35]. It started from building a digital currency system on the P2P network that provides the immutability of data without depending on a central organization [24]. Technologies like smart contracts enabled blockchain to store more complex structures such as a code and its execution history [6], and the blockchain became a tool for building a general-purpose application on a decentralized network.

Another important decentralized technology is decentralized storage [3], such as IPFS [29] and Swarm [10]. It is a distributed data storage on a decentralized network built on blockchain technology. Unlike the traditional distributed network, it uses the content of the file, e.g., file hash, instead of the node's IP address to locate the file. It makes it easier to integrate the storage network into the blockchain network that consists of anonymous nodes. It uses unused hardware spaces of people to store the data and usually uses a cryptocurrency incentive system to let them manage the data. It is becoming an alternative method to centralized cloud storage for storing large files since its decentralized structure helps protect attacks and decreases data management costs.

Our work utilizes a decoupled approach to using the blockchain network and decentralized storage together for efficiently supporting cooperative 3D modeling.

2.2 Cooperative 3D modeling

There have been prior approaches to enable collaboration in 3D modeling in different directions. MeshGit proposes diffing and merging algorithms for 3D polygonal meshes to enable version control of 3D models [8]. SceneGit constructs a version control system for the various components of the 3D scene, such as shapes, materials and textures, which is based on a heuristic method that is robust and efficient for the large 3D scenes [7]. MeshHisto supports sharing and merging mesh version histories for real-time 3D modeling collaboration [31].

Pixar proposes a format (language) for describing complex 3D scenes, which uses a layered structure for enabling collaboration [28]. CoMaya shows that we can connect users of the 3D modeling tool without modifying the source code of the tool [1]. Conflict resolution of different 3D model versions is addressed by separating conflict into multiple types [9]. Omniverse constructs a cloud-based collaboration modeling platform that provides data exchange between multiple kinds of 3D modeling tools to allow the designers to use multiple tools [26].

These prior approaches focus on developing management and the exchange of 3D modeling data to connect the 3D modeling tools through the network. However, these techniques paid less attention to the security of the data and intellectual property. Since the 3D model has become a valuable asset, its security should be considered critical for a database. These previous techniques use a centralized network to connect the 3D modeling tools and manage 3D modeling data, which is vulnerable to network attacks, physical disasters, and internal tempering. Instead, we propose a decentralized network powered by blockchain and decentralize storage to provide state-of-the-art security in practice.

2.3 Decentralized version control

There are approaches applying blockchain technology to version control. Gitchain [30] and Mango [4] are two implementations of using the blockchain network as a backend for Git, a version control system for code.

Similar to code cooperation, cooperative 3D modeling or database also relies on a version control algorithm. However, it is a more challenging problem due to the need for much larger and more frequent real-time data transactions. Therefore, we propose a specific format (called 'mesh page') for decreasing the data size by adopting 3D data compression on the difference between two versions of the 3D model. The simple technique reduces the network load and helps improve the performance of uploading and downloading the 3D models.

2.4 Decentralized technologies in computer graphics

There are some computer graphics techniques incorporated into decentralized technologies. Besançon et al. [5] proposed a data representation of 3D assets for enabling blockchain-based data exchange in 3D applications such as video games. OTOY [27] and Golem [14] proposed a distributed GPU rendering system on the blockchain, which allows the participants to perform rendering tasks submitted

by others (end-users) and receive the proper amount of cryptocurrency. OTOY [27] pointed out that incorporating blockchain technology would help to protect intellectual rights.

Departing from those prior approaches, our platform tracks the history of the cooperative modeling process. It also provides matching 3D models, which can protect the intellectual rights of cooperative 3D property and authenticate data sources.

3 Overview

To realize a secure database for 3D models and intellectual properties, we consider recent advances in the distributed system field. Blockchain is a distributed append-only ledger, i.e., a database of transactions on a peer-to-peer network. It stores the transactions on distributed network nodes with append-only (immutable, once written) modification records. Its security mechanism is practically unbreakable. However, blockchain is appropriate for managing small data such as numbers and text rather than managing large data such as a 3D model due to the synchronization cost. Therefore, we use the blockchain network to manage the 3D models' intellectual properties and indices. Also, to store the actual 3D data, we use a decentralized storage network that is appropriate for storing large data on a decentralized network.

Fig. 1 illustrates an overview of our platform with fundamental operation blocks. The platform contains three components, the local client, a blockchain network (called 'index blockchain network'), and a decentralized storage network (called 'decentralized mesh network'). The local client provides basic UI for editing, uploading, and downloading models for different applications. The index blockchain network provides efficient and immutable management of the intellectual property and data structure. The decentralized mesh network can safely store the large-size 3D models by distributing the data to the decentralized network.

We explain four different applications that can be supported by MeshChain:

- **Data registering and storing.** Users can update the data to the decentralized mesh network and the intellectual information to the index blockchain network for the purpose of 3D property protection.
- **Intellectual property management.** Since we record every submission on the blockchain, we use that data to calculate the contribution of each author.
- **Data authentication.** Users can download the models to authenticate a model. The local client calculates and shows the geometry and visual similarities for matching models. (Fig. 3)
- **Cooperative modeling.** For committing a model, local clients calculate mesh pages representing the difference between the prior and current commits (Section 4) and commit them as a new, single transaction. For checkout, the local client queries the latest block from the index blockchain network

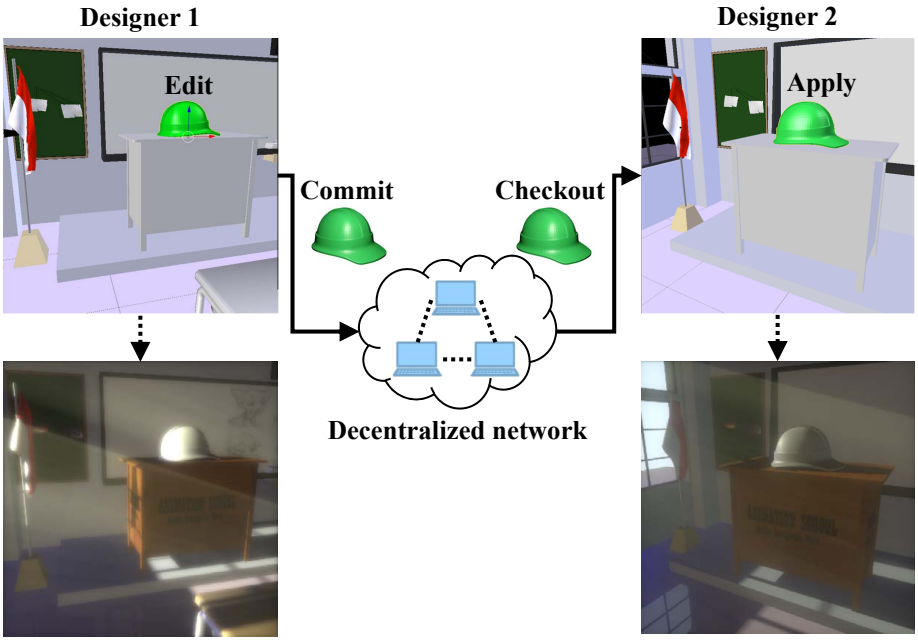

Fig. 2. Example of cooperative modeling. Designer 1 adds a model (colored by green) on the scene and submits its commit to the network. Designer 2 can then perform checkout to download the model added by designer 1. So, both designers can share and render the model.

for the latest version or trackbacks the chain to get different branches. The mesh is downloaded from the decentralized mesh network according to the storage address. We also construct 'commit incentive', a cryptocurrency paid to the author when the author submits a commit. It encourages users to participate in the modeling project and improves the 3D model. Such a feature might be especially useful in community-driven projects. (Fig. 2)

4 Cooperative modeling

In this section, we describe main components of our method for enabling cooperative modeling based on the blockchain.

4.1 Commit and checkout

Various applications mentioned above require two main operations: the commit and checkout of 3D models. As illustrated in Fig. 1, the local client, index blockchain network, and decentralized mesh network cooperates to accomplish the operations of commit and checkout.

For the commit operation, users can import and edit 3D models using 3D modeling tools, such as Blender, 3ds Max, and Maya. The local client then performs the creation and submission of a transaction. Before creating a transaction, the local client calculates the difference between the current mesh and the previous commit, called 'mesh page' for efficient communication to the blockchain network, and submits it to the decentralized mesh network. Given the storage address on the decentralized mesh network, the local client creates a transaction that records the previous commit ID, the storage address, tags (keywords for searching), the author's address (the author's blockchain account), date, and additional data.

The local client submits the transaction to nearby 'full nodes' of the blockchain network. Full nodes are the blockchain network nodes that are maintaining the entire blockchain in their local storage by creating the blocks and verify incoming transactions and blocks. The full nodes broadcast the new transaction to each other and verify it. Several verified transactions are merged into a new block, which is linked to the end of the blockchain as an immutable record.

For the checkout, the local client first queries the commit information such as storage address from the index blockchain network. Then the local client downloads actual geometry data from the decentralized mesh network and provides the expected mesh to the user.

4.2 Mesh page

In order to reduce the network load and network operation overhead, we propose a 3D-specific strategy called 'mesh page'.

When we submit a commit, we store only the difference between the mesh of the current commit and the mesh of the previous commit, instead of storing the whole mesh. In addition, the time of storing the data on the decentralized storage increases almost linearly for the small number of triangles, but it becomes larger than expected if the number of triangles is very large (Table 2). So we partition the set of triangles into multiple pieces and then compress each piece by using the 3D compression algorithm such as OpenCTM [13]. We show that how much performance boosts these steps can make in Section 6.1.

Specifically, we stored each piece to the decentralized storage and its address to the index blockchain network and continue to process the next piece of the 3D model. In this iterative process, we measured the total time spent on storing the data on the decentralized storage network and the separate time & total time spent on storing its address on the blockchain network. We can observe that the blockchain's overhead is significantly larger than the overhead of storing the data on the decentralized storage network (Fig. 5).

Specifically, when storing the (compressed) pieces of the 3D model, we first store all the pieces to the storage network, and then merge their addresses into one string and store that string to the blockchain at once, to reduce the overhead of accessing the blockchain network.

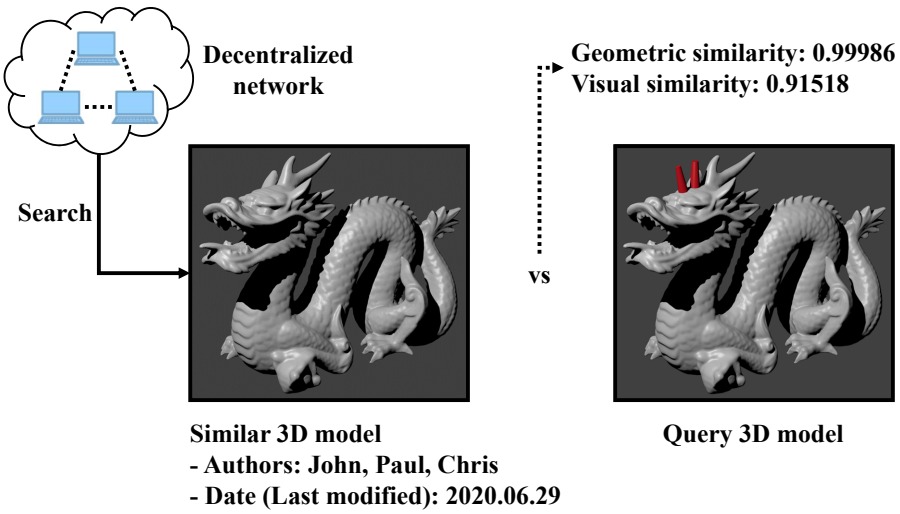

Fig. 3. Example of data authentication. Since the decentralized network securely holds the 3D data and important information (e.g., author, date, etc.), a designer can search 3D models that are similar to the query model. The designer can also find which part of a similar 3D model infringes the intellectual right by comparing those models.

5 Intellectual property management

Calculating and managing intellectual property is an important challenge in software management. Similarly, cooperative 3D modeling and public 3D model databases face the challenge of calculating the contributions among the participants. While the demands for cooperative modeling and large-scale public 3D database keep increasing, there have been few studies on the intellectual management of the 3D property. Besides the evaluation metric, the transparency and consistency of intellectual management are also important, especially for the cooperation of communities.

5.1 Mesh incentive

Inspired by SLOC (Source Lines of Code) [25,19], the number of newly modified triangles (let's call it 'effective triangles') can be a unit of measurement for the work effort of modeling. Modern blockchains such as Ethereum [6] supports stacking the cryptocurrency on the network via smart contract [11]. Based on these ideas, we propose 'mesh incentive', an incentive distributed to the contributors of a 3D modeling project, which can be considered as a metric of 3D modeling contribution.

Simply in our work, mesh incentive, I, regards the number of effective triangles as the contribution:

$$I \propto |T_{\text{eff}}|, \tag{1}$$

where T_{eff} is a set of effective triangles. T_{eff} is defined as:

$$T_{\text{eff}} = T \setminus \bigcup_{k=1}^{n} T_k, \tag{2}$$

where T is a set of all mesh triangles modified (added & removed) by a user and $T_1, \cdots, T_n$ are the modified triangles in each past commit operation. Since T_{eff} excludes all the triangles submitted in the past, it prevents the user to abuse the incentive system, e.g., submitting the same data repeatedly to get a lot of incentive.

Suppose that the incentive supply, say r, is stacked on the blockchain. We need to guarantee $I < r$ so that the platform will not run out of incentive for new contributions. Additionally, we also have to balance the distribution of incentives to the participants of the project. To solve these issues, we set I like this:

$$I = \frac{|T_{\text{eff}}|}{|T|} \times \frac{r}{u+1}, \tag{3}$$

where r is the remaining incentive supply, u is the current number of participants in the modeling project. (In our implementation, we count u by just considering everyone who commits at least once to the network as an author.) Since $|T_{\text{eff}}| \leq |T|$ and $u+1 > 1$, we have $I < r$. As a result, a user can always get an incentive if the user adds at least one new triangle to the mesh. After the blockchain gives I to the user, the supply r will decrease to $r - I$. Our implementation (Section 6) uses this version (Equation (3)) of the mesh incentive.

We can extend this formula to handle the 'importance' of each triangle. There have been several methods for measuring the importance of each triangle, called 'mesh saliency' [22,21]. Let $s(t)$ the saliency (importance) of the triangle $t \in T$. We regard the saliency as 'weight' of each triangle in T_{eff}:

$$I = \frac{\sum_{t \in T_{\text{eff}}} w(t)}{|T|} \times \frac{r}{u+1}. \quad \left(w(t) = \frac{s(t)}{\sum_{t \in T} s(t)} \right) \tag{4}$$

Since $\sum_{t \in T_{\text{eff}}} w(t) \leq \sum_{t \in T_{\text{eff}}} 1 \leq |T_{\text{eff}}|$, we still have $I < r$.

5.2 Data authentication

Since the Internet became the most popular repository for digital data, the data authentication problem has been getting increasing attention. 3D models and designs are vulnerable to tampering, plagiarism, or piracy while being propagated. The important data, such as the author information associated with the models, can also be easily modified. We thus suggest using a transparent, decentralized, and immutable platform to register and authenticate important 3D properties.

Since 3D models are stored immutably on our network, we can apply the similarity of 3D models to compare the relationship between two models, such as the model stored in the network and the model which looks similar to it (Fig. 3). Measuring visual and geometric similarity has been an active research

topics [12,15], but we adopt simple approaches as a proof-of-concept for our approach. We may have to adopt more complex approaches such as learning-based geometrical similarity [12] to implement real-world application of our idea. In our approach, geometry similarity simply computes the ratio of common triangles between the two models. Visual similarity measures a visual matching rate. Visual clue provides high-level information about the appearance of the 3D model and retains robustness even if the low-level geometry details are changed.

Models may have different geometry details or formats, but they can be visually similar if they stem from the same design. In this regard, the visual similarity provides a metric to match models from the design perspective.

6 Results

In this section, we discuss the results of our approach with various tested applications built on top of our prototype.

We implemented our approach using Ethereum blockchain [6] and Swarm decentralized storage [10]. The client, written in Kotlin, communicates with Blender modeling tool via RPC (Remote Procedure Call), Ethereum via Web3j [23], and Swarm via HTTP (Fig. 4). The client reads/writes the information from those tools and performs 3D operations. For example, when the user creates the commit, it reads the 3D model from Blender, calculates the mesh page, stores the data on Swarm, calculates the mesh incentive, and creates a transaction to Ethereum. Then the smart contract (executable code on the blockchain) stores the commit address on the Ethereum network and pays the mesh incentive to the author. The client provides calculating the visual similarity of the 3D model of a commit and the external 3D model. We used match3d library [16] for performing visual similarity, and used OpenCTM [13] for 3D compression. The code and demonstration video of our platform are available at https://github.com/Avantgarde95/MeshChain-publish. The video is also available at https://youtu.be/xLU79JfRdbQ.

We built a small test network on the servers connected by LAN, where each server runs its own instances of Ethereum and Swarm clients. For the performance analysis, we used the localhost network (i.e., single computer) to minimize the overhead of the network.

6.1 Decentralized storage and mesh page

We compare the performance of using different techniques of the proposed platform when conducting commit/checkout of a 3D model.

In Table 1, the first approach (Blockchain) submits the mesh data directly to the blockchain network. The second approach (+ Storage) stores the mesh data on the integrated decentralized storage network, and the blockchain holds its storage address instead. The third approach (+ Mesh page) additionally applies the mesh difference and mesh compression of our mesh page technique.

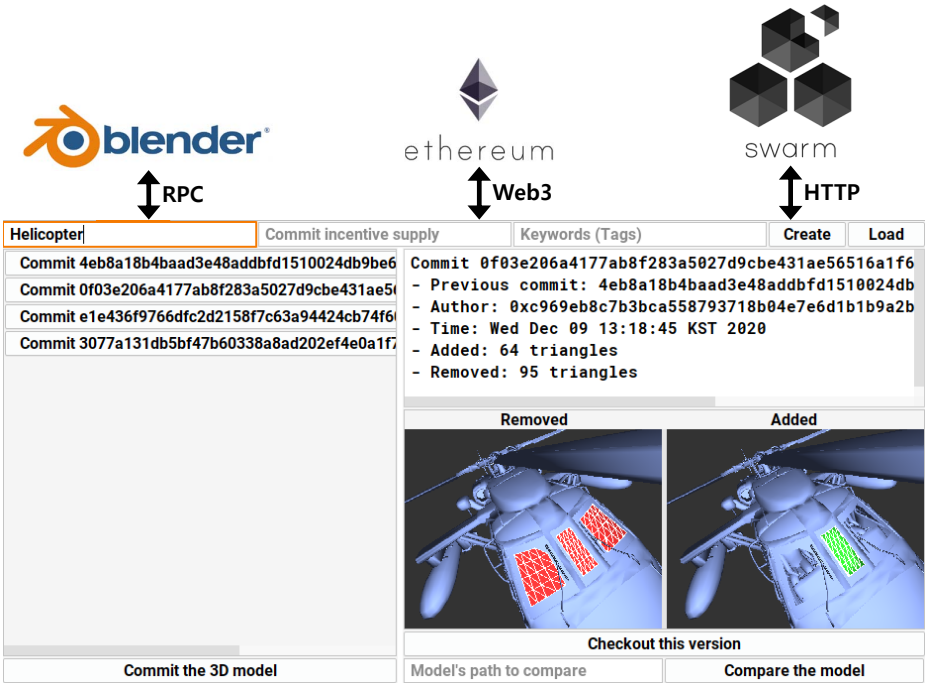

Fig. 4. Our prototype is built on Blender modeling tool, Ethereum blockchain, and Swarm decentralized storage. Applications introduced in this paper are realized via the client code and the smart contract code.

The commit time is decomposed into two parts: the time of committing to the decentralized storage network and to the blockchain network.

Since the blockchain network has to synchronize and store a complete copy of the blocks in every full nodes' local storage, the blockchain usually sets the synchronization cost (called 'GAS'), whose value is related to the size of data. The large data size of the 3D model can cause the blockchain's synchronization cost to exceed an internal limit (called 'GAS limit') set by the network initiator; in the real world, the GAS limit is about 8,000,000 gwei, and in our test network, we set the limit to 1,000,000,000 gwei for testing bigger models.

As shown in Table 1, unless the 3D model is very small, using only the blockchain itself is unable to support 3D models. By using decentralized storage together, we can handle such models. Furthermore, compressing the mesh reduces the storage requirement by a factor of 3 to 40 times and gives the performance boost for the large models. Note that in this scenario (submitting a new model), mesh difference does not help boost the performance, so for the small models (< 10K triangles) the processing time for commit does not decrease.

The mesh difference technique in our mesh page shows its power when we modify the already submitted model. In Fig. 6, we assume that a 3D model of

69K triangles is stored on the network, and an author wants to modify some triangles of the model. The mesh page technique clearly reduces the operation times, especially the checkout time. The commit time decreases by 3.6% on average, and the checkout time decreases by 90.2% on average. The commit time decreases relatively less because of the blockchain's overhead of verifying and applying the new data.

Partitioning the mesh. We now check whether applying the mesh separation in our mesh page gives us a performance boost or not. As a simple test, we separate the mesh into just two pieces. We measure the time to store the mesh on the network Table 2. We can clearly see that the mesh separation and compression give a performance boost, especially for large models.

7 Conclusion and discussion

We have proposed a secure 3D model and intellectual property management platform powered by blockchain and decentralized storage techniques. The platform can support various applications, including cooperative modeling, intellectual property management, 3D data authentication, and 3D data protection. More importantly, both the 3D data and the intellectual property can be protected by a decentralized and immutable system. The mesh page technique reduces the workload of the system.

There are some limitations in the current implementation, which also enlightens possible future works. Currently we don't apply any kind of encryptions to the 3D models. Adapting secure rendering through remote rendering [20] to our platform will enable performing confidential 3D projects on our platform. Handling not only the triangles but also the material information such as texture and color will enable more flexible cooperative modeling. More sophisticated geometry feature extraction techniques can improve the performance of geometry matching.

Appendix

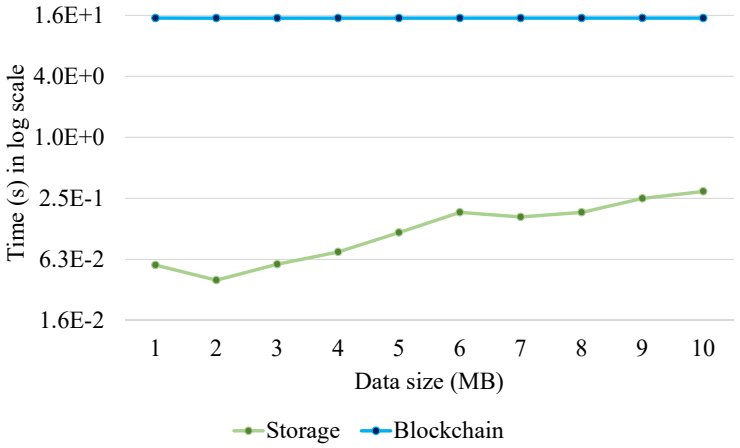

Fig. 5. Time of storing the data on the Swarm [10] storage and storing its storage address on the Ethereum [6] blockchain. The graph shows that the blockchain overhead is significantly larger than the storage overhead. The average time is reported out of 10 different tests.

	Blockchain	+ Storage	+ m.d. & m.c.
Model (# of Δ)	commit (s) checkout (s) storage (MB)	commit (s) checkout (s) storage (MB)	commit (s) checkout (s) storage (MB)
Cube (12)	30.104 0.149 0.001MB	30.234 0.184 0.001MB	30.538 0.137 0.0003MB
Table (1,267)	N.A. N.A. N.A.	30.120 0.159 0.176MB	30.113 0.098 0.009MB
Copter (46,703)	N.A. N.A. N.A.	31.667 0.951 6.356MB	31.745 0.196 0.324MB
House (427,647)	N.A. N.A. N.A.	37.404 7.797 61.959MB	31.730 1.001 1.568MB

Table 1. The performances of storing and restoring different models by using only the blockchain network, additionally using the decentralized storage network, and using the mesh difference and mesh compression of our mesh page method.

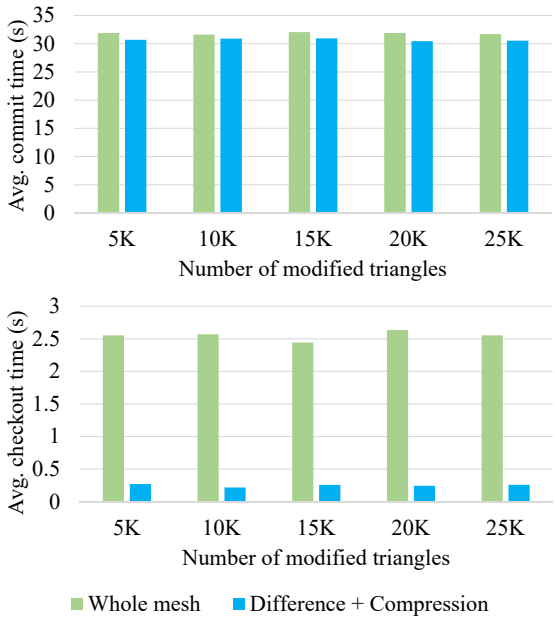

Fig. 6. Comparison in terms of commit and checkout times between using the whole mesh and using mesh pages, when we modify different portions of a bunny model consisting of 69 K triangles. The average is computed out of four different tests.

# of triangles	427,647	2,000,000
Store the model to the storage	6.5s	103.4s
Store the model's address to the blockchain	15.1s	15.0s
Whole process	21.5s	118.4s
Size of each piece	213,824	1,000,000
Store each piece to the storage	2.5s	33.7s
Store the pieces' address to the blockchain	15.0s	15.0s
Whole process	**20.0s**	**82.4s**
Compress & store each piece to the storage	1.2s	3.1s
Store the pieces' address to the blockchain	15.0s	15.0s
Whole process	**17.4s**	**21.1s**

Table 2. Results w/ and w/o using separation & compression of mesh page.

References

1. Agustina, Fei Liu, Steven Xia, Haifeng Shen, and Chengzheng Sun. Comaya: Incorporating advanced collaboration capabilities into 3d digital media design tools. page 5–8, 2008.
2. Adam W Bargteil and Elaine Cohen. Animation of deformable bodies with quadratic bézier finite elements. *ACM Transactions on Graphics (TOG)*, 33(3):1–10, 2014.
3. Nazanin Zahed Benisi, Mehdi Aminian, and Bahman Javadi. Blockchain-based decentralized storage networks: A survey. *Journal of Network and Computer Applications*, 162:102656, 2020.
4. Alex Beregszaszi. Mango. https://github.com/axic/mango, 2016.
5. Léo Besançon, Catarina Ferreira Da Silva, and Parisa Ghodous. Towards blockchain interoperability: Improving video games data exchange. *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 81–85, 2019.
6. Vitalik Buterin. Ethereum: A next-generation smart contract and decentralized application platform, 2014. Accessed: 2016-08-22.
7. Edoardo Carra and Fabio Pellacini. Scenegit: a practical system for diffing and merging 3d environments. *ACM Transactions on Graphics (TOG)*, 38(6):1–15, 2019.
8. Jonathan D. Denning and Fabio Pellacini. Meshgit: Diffing and merging meshes for polygonal modeling. *ACM Trans. Graph.*, 32(4):35:1–35:10, July 2013.
9. Jozef Doboš and Anthony Steed. 3d diff: An interactive approach to mesh differencing and conflict resolution. In *SIGGRAPH Asia 2012 Technical Briefs*, SA '12, pages 20:1–20:4, New York, NY, USA, 2012. ACM.
10. Ethereum Foundation. Swarm: Storage and communication for a sovereign digital society. https://swarm.ethereum.org/, 2017.
11. Ethereum Foundation. Solidity. https://docs.soliditylang.org/, 2016.
12. Takahiko Furuya and Ryutarou Ohbuchi. Deep aggregation of local 3d geometric features for 3d model retrieval. In *BMVC*, volume 7, page 8, 2016.
13. Marcus Geelnard. Openctm. http://openctm.sourceforge.net/, 2009.
14. Golem Factory GmbH. The golem project. Technical report, 2016.
15. Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *European conference on computer vision*, pages 241–257. Springer, 2016.
16. Ryan Henderson. match3d. https://github.com/ascribe/match3d, 2016.
17. Yuchi Huo, Rui Wang, Ruzahng Zheng, Hualin Xu, Hujun Bao, and Sung-Eui Yoon. Adaptive incident radiance field sampling and reconstruction using deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 39(1):1–17, 2020.
18. Evangelos Kalogerakis. Session details: Learning geometry. *ACM Transactions on Graphics (TOG)*, 37(6), 2018.
19. Tim Kelleher. Five core metrics-the intelligence behind successful software management. *Software Quality Professional*, 6(2):44, 2004.
20. David Koller, Michael Turitzin, Marc Levoy, Marco Tarini, Giuseppe Croccia, Paolo Cignoni, and Roberto Scopigno. Protected interactive 3d graphics via remote rendering. *ACM Trans. Graph.*, 23(3):695–703, August 2004.
21. Guillaume Lavoué, Frédéric Cordier, Hyewon Seo, and Mohamed-Chaker Larabi. Visual attention for rendered 3d shapes. In *Computer Graphics Forum*, volume 37, pages 191–203. Wiley Online Library, 2018.

22. Chang Ha Lee, Amitabh Varshney, and David W. Jacobs. Mesh saliency. *ACM Trans. Graph.*, 24(3):659–666, July 2005.
23. Web3 Labs Ltd. Solidity. https://www.web3labs.com/web3j-sdk, 2018.
24. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system," http://bitcoin.org/bitcoin.pdf, 2009.
25. Vu Nguyen, Sophia Deeds-Rubin, Thomas Tan, and Barry Boehm. A sloc counting standard. In *Cocomo ii forum*, volume 2007, pages 1–16. Citeseer, 2007.
26. NVIDIA. Omniverse. https://developer.nvidia.com/nvidia-omniverse, 2019.
27. OTOY. Render token (rndr) whitepaper. Technical report, 2017.
28. Pixar. Usd (universal scene description). https://graphics.pixar.com/usd/docs/index.html, 2016.
29. Protocol Labs. Ipfs(interplanetary file system): Content addressed, versioned, p2p file system. https://ipfs.io/, 2015.
30. Yurii Rashkovskii. Gitchain. http://gitchain.org/, 2020.
31. Gabriele Salvati, Christian Santoni, Valentina Tibaldo, and Fabio Pellacini. Meshhisto: Collaborative modeling by sharing and retargeting editing histories. *ACM Trans. Graph.*, 34(6):205:1–205:10, October 2015.
32. Melanie Swan. *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.", 2015.
33. Sarah Underwood. Blockchain beyond bitcoin, 2016.
34. Sung-eui Yoon. *Rendering.* 2018. 'First edition'.
35. Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375, 2018.
36. Gaspard Zoss, Derek Bradley, Pascal Bérard, and Thabo Beeler. An empirical rig for jaw animation. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.