DOI: 10.5281/zenodo.17447452
Link: https://zenodo.org/records/17447452

3D VISUALIZATION AND MODELING OF HUMAN ORGANS USING INNOVATIVE TECHNOLOGIES

Maftunakhon A. Mahmudova
(Belarus–Uzbekistan Intersectoral Institute of Applied Technical Qualifications, Tashkent, Uzbekistan)

Abstract - *The article analyzes the scientific–methodological foundations and modern technological approaches to three-dimensional (3D) visualization and modeling of human organs in the medical field. It explores the processes of processing DICOM-formatted CT/MRI data, performing segmentation, and creating 3D anatomical models using environments such as Blender, 3D Slicer, Unity, and Unreal Engine. The proposed methodology is aimed at improving medical diagnostics, surgical planning, and clinical analysis processes. Furthermore, the article substantiates the scientific and practical significance of developing a 3D anatomy system with an Uzbek-language interface based on local medical terminology.*

Keywords: *3D anatomy, medical visualization, DICOM, computer graphics, CT, MRI, modeling, VR/AR technologies.*

INTRODUCTION

In the 21st century, the integration of innovative technologies into medical education and practice has become a matter of particular importance. The use of 3D models in studying anatomy strengthens the knowledge of medical students, enhances doctors' practical skills, and simplifies clinical diagnostic processes. Although a number of 3D anatomy simulators exist worldwide, most of them are in foreign languages, creating linguistic, methodological, and resource limitations for Uzbekistan's education system and healthcare professionals. Therefore, there is a growing need to develop a 3D anatomy simulator in the Uzbek language with a user-friendly interface based on national medical terminology.

Globally, platforms such as Visible Body, Anatomage Table, and 3D4Medical (Complete Anatomy) are widely used. These tools allow users to view anatomical structures interactively, isolate organ layers, and simulate physiological processes. However, most of these programs are only available in English, require expensive licenses, and are not adapted to local educational standards — factors that significantly limit their use in Uzbekistan.

For example, Visible Body is a comprehensive 3D visual education platform covering all systems of human anatomy, allowing students and doctors to model anatomical, physiological, and pathophysiological processes with high visual accuracy. The program provides 3D rotation, dissection, isolation, and animation functions, with a license cost of USD 200–500.

Anatomage Table is a physical 3D interactive anatomy table that enables virtual dissection based on real human CT and MRI data, allowing layered visualization of organs. Its system price ranges between USD 50,000 and 80,000.

Meanwhile, 3D for Medical, developed in collaboration with Apple, offers a high-quality anatomy simulator featuring layered dissection, VR/AR-based training, and clinical simulation capabilities. Its advantages include high visual fidelity, pedagogical interactivity, and convenience for surgical education, with a license cost of USD 100–150.

In Uzbekistan, however, factors such as the language barrier, discrepancies in local anatomical terminology, lack of integration with CT/MRI data, and high license costs restrict the use of such platforms in medical institutions. Although global systems like Visible Body, Anatomage Table, and 3D4Medical have proven effective internationally, adapting them to the national education system

remains crucial. Developing a locally designed 3D anatomy simulator tailored to Uzbekistan's linguistic and educational context could mark a new stage in the country's medical education, enhance doctors' practical training, and strengthen the scientific potential of medical innovations.

LITERATURE REVIEW

The creation of a 3D anatomy simulator in medical education requires the comprehensive application of advanced digital technologies. This process involves several key stages, including interactive modeling, processing of real diagnostic data, and optimization of the user interface.

1. Core Development Environment. The main part of the 3D anatomy simulator is developed using the Unity or Unreal Engine platforms, which serve as the primary physics and graphics engines. These environments enable highly realistic visualization, interactive manipulation (such as isolating, rotating, and enlarging organs), and full VR/AR support.

Moreover, these systems allow developers to create localized user interfaces (UI) in various languages, including Uzbek, thus laying the foundation for an interactive, native-language medical education platform.

2. Creation of 3D Anatomical Models. The 3D anatomical models required for the simulator are designed using Blender and 3D Slicer software and are processed based on medical tomography data. The 3D Slicer program enables segmentation of CT/MRI data in DICOM format, allowing each organ or tissue layer to be identified separately. After segmentation, the data are imported into Blender, where they undergo processes such as refinement, smoothing, and texturing. As a result, high-precision anatomical models are produced for integration into the simulator.

3. Integration of Medical Diagnostic Data. To upload and manage diagnostic data, the system employs the Orthanc DICOM Server, which provides secure storage, management, and export of radiological information to a 3D environment. This allows users to perform visual analyses based on anonymized patient data, study the structure of organs in detail, and strengthen their clinical diagnostic skills. Such integration bridges the gap between theoretical knowledge and practical experience, enhancing the realism of medical training.

4. Web-Based Access through WebGL. The WebGL-based online version of the simulator allows users to explore 3D anatomy models interactively through an internet browser, without installing additional software. This feature significantly improves accessibility, making it possible for medical students and practitioners to study anatomical structures anytime and anywhere.

METHODOLOGY

Simulation of Physiological Heart Function. A Unity C# script is used to control the rhythmic contraction and expansion of a 3D heart model based on sinusoidal functions. This approach replicates the physiological heartbeat frequency in a digital environment. Below is the provided code snippet:

```
{
  public float bpm = 70f;
  public float amplitude = 0.04f;
  public Vector3 scaleAxis = new Vector3(1, 1, 1);
  private Vector3 baseScale;
  private float phaseOffset;
  void Start ()
  {
    baseScale = transform.localScale;
    phaseOffset = Random.Range(0f, 2f * Mathf.PI);
  }
  void Update ()
  {
    float beatsPerSec = bpm / 60f;
```

```
        float t = Time.time * beatsPerSec * 2f * Mathf.PI + phaseOffset;
        float s = 1f + amplitude * Mathf.Sin(t);
        transform.localScale = Vector3.Scale(baseScale * s, scaleAxis);
    }
    public void SetBPM(float newBpm)
    {
        bpm = Mathf.Clamp(newBpm, 20f, 220f);
    }
}
```

Using this script, the 3D heart model performs periodic expansion and contraction similar to a real heartbeat.

- A rate of 60–90 bpm simulates a normal, healthy rhythm.
- Rates above 100 bpm model tachycardia (abnormally fast heartbeat).
- Rates below 40 bpm model bradycardia (abnormally slow heartbeat).

ANALYSIS AND RESULTS

Visualization of Blood Flow. The following BloodFlowController.cs script visually simulates blood flow inside the 3D heart model using ParticleSystem objects in Unity to animate the movement of red blood cells:

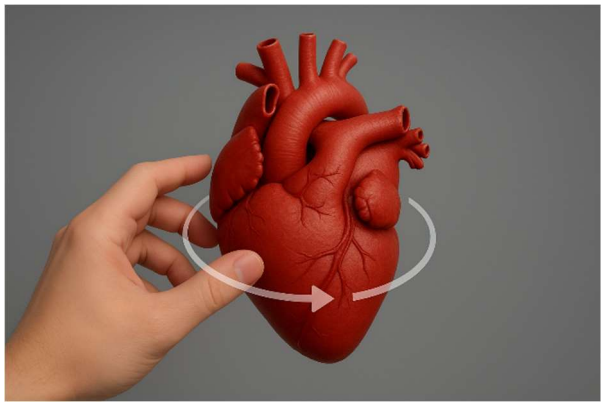
```
{
    public ParticleSystem bloodParticles;
    public float flowRate = 1.0f;
    void Start()
    {
        if (bloodParticles == null) bloodParticles = GetComponent<ParticleSystem>();
    }
    void Update()
    {
        var emission = bloodParticles.emission;
        emission.rateOverTime = Mathf.Clamp(50f * flowRate, 0f, 1000f);
    }
    public void SetFlowRate(float rate)
    {
        flowRate = Mathf.Clamp01(rate);
    }
}
```

Interactive Control of the Heart Model

The HeartInteractor.cs script allows users to interact with the 3D heart model through mouse-based actions — rotation, zooming, and panning — providing a fully interactive learning experience.

```
{
    public Transform target;
    public float rotationSpeed = 120f;
    public float zoomSpeed = 2f;
    public float minZoom = 0.5f;
    public float maxZoom = 3.0f;
    private Vector3 prevMousePos;
    private float currentZoom = 1f;
    void Start()
    {
        if (target == null) target = this.transform;
        currentZoom = target.localScale.x;
```

```
    }
    void Update()
    {
        if (Input.GetMouseButton(1))
        {
            Vector3 delta = Input.mousePosition - prevMousePos;
            float yaw = -delta.x * rotationSpeed * Time.deltaTime;
            float pitch = delta.y * rotationSpeed * Time.deltaTime;
            target.Rotate(Vector3.up, yaw, Space.World);
            target.Rotate(Vector3.right, pitch, Space.World);
        }
        float scroll = Input.GetAxis("Mouse ScrollWheel");
        if (Mathf.Abs(scroll) > 0.0001f)
        {
            currentZoom = Mathf.Clamp(currentZoom + scroll * zoomSpeed, minZoom, maxZoom);
            target.localScale = Vector3.one * currentZoom;
        }
        prevMousePos = Input.mousePosition;
    }
}
```

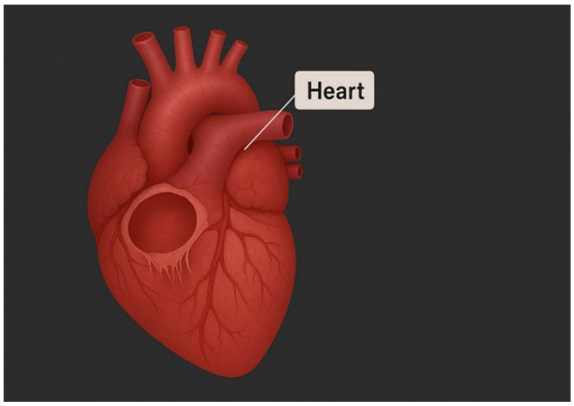


Picture -1. 3D anatomical model of the human heart with interactive rotation visualization

With the HeartInteractor component, the user can view the anatomical model in 360°, perform detailed observation, and conduct interactive analysis of different structures. The process of adding labels (names) to anatomical parts and visualizing them in 3D space is implemented as follows:

```
{
    public string labelText = "Unknown Part";
    public GameObject labelPrefab
    private GameObject labelInstance;
    void Start()
    {
        if (labelPrefab != null)
        {
            labelInstance = Instantiate (labelPrefab, transform.position + transform.up * 0.02f, Quaternion.identity);
            labelInstance.transform.SetParent(transform, true);
            var textComponent = labelInstance.GetComponentInChildren<TextMeshPro>();
            if (textComponent != null)
                textComponent.text = labelText;
```

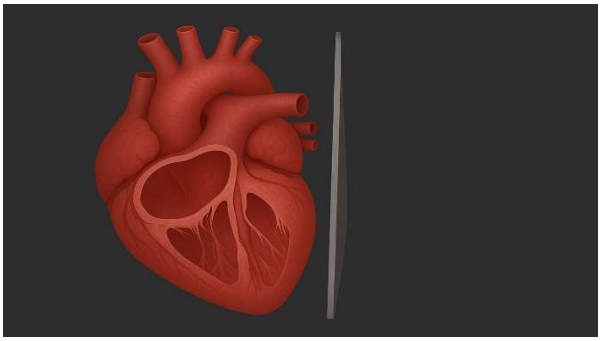
```
            }
            else
            {
                Debug.LogWarning($"[{name}] Label prefab biriktirilmagan!");
            }
        }
        public void SetLabel(string text)
        {
            labelText = text;

            if (labelInstance != null)
            {
                var textComponent = labelInstance.GetComponentInChildren<TextMeshPro>();
                if (textComponent != null)
                    textComponent.text = labelText;
            }
        }
    }
```



Picture -2. 3D anatomical model of the human heart with labeled identification.

Labeling and 3D visualization of the model, and performing a virtual cross-section of the 3D structure:

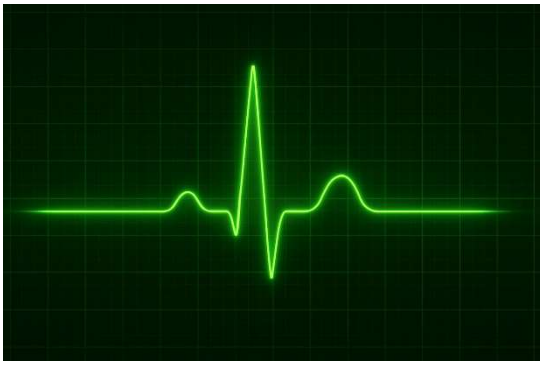
```
    {
        public Material sliceMaterial;
        public Transform clipPlaneTransform;
        void Update ()
        {
            if (sliceMaterial == null || clipPlaneTransform == null)
                return;
            Vector3 planeNormal = clipPlaneTransform.up;
            Vector3 planePos = clipPlaneTransform.position;
            float d = -Vector3.Dot(planeNormal, planePos);
            Vector4 plane = new Vector4(planeNormal.x, planeNormal.y, planeNormal.z, d);
            sliceMaterial.SetVector("_ClipPlane", plane);
        }
    }
```



Picture 3. The virtual cross-sectional effect of the 3D model

The EKGSimulator.cs script produces a realistic electrocardiogram (ECG) signal that emulates the activity of heartbeats:

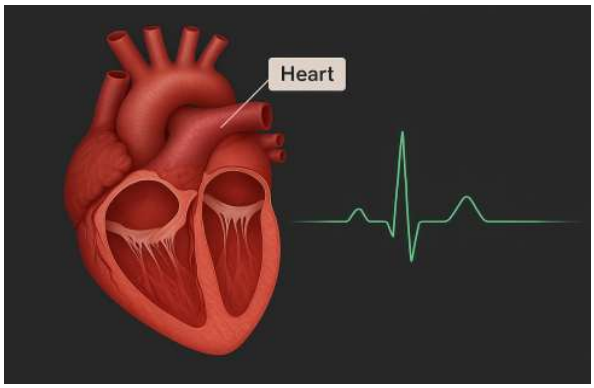
```
{
   public float bpm = 70f;
   public LineRenderer lineRenderer;
   public int points = 500;
   private float t = 0f;
   void Start ()
   {
      if (lineRenderer == null)
         lineRenderer = GetComponent<LineRenderer> ();
      lineRenderer.positionCount = points;
   }
   void Update ()
   {
      float beatsPerSec = bpm / 60f;
      t += Time.deltaTime;

      for (int i = 0; i < points; i++)
      {
         float x = i / (float)points * 10f;
         float val = Mathf.Sin((t + i * 0.02f) * beatsPerSec * 2f * Mathf.PI) * 0.5f;
         val += Mathf.Exp(-Mathf.Pow((i % 50 - 25) / 5f, 2)) * 1.2f * Mathf.Sin((t + i) * 10f);
         lineRenderer.SetPosition(i, new Vector3(x, val, 0));
      }
   }
   public void SetBPM(float newBpm)
   {
      bpm = Mathf.Clamp(newBpm, 30f, 220f);
   }
```



Picture 4. The signal of EKG ang signal of UI

By combining the results, we obtain the following:

```
{
   [Header ("Core Systems")]
   public HeartBeat heartBeat;
   public BloodFlowController bloodFlow;
   public EKGSimulator ekg;

   [Header ("UI Components")]
   public Slider bpmSlider;
   public Slider flowSlider;
   public TextMeshProUGUI bpmText;

   void Start ()
   {
            bpmSlider.onValueChanged.AddListener(OnBPMChanged);
      flowSlider.onValueChanged.AddListener(OnFlowChanged);

      bpmSlider.value = heartBeat.bpm;
      flowSlider.value = bloodFlow.flowRate;
      bpmText.text = $"{heartBeat.bpm: F0} BPM";
   }
   void OnBPMChanged(float val)
   {
      heartBeat.SetBPM(val);
      ekg.SetBPM(val);
      bpmText.text = $"{Mathf.RoundToInt(val)} BPM";
   }
 void OnFlowChanged(float val)
   {
      bloodFlow.SetFlowRate(val);
   }
}
```



Picture 5. The complete version of the UIManager.cs script

CONCLUSION

The results of this study make it possible to develop an innovative methodology for creating 3D anatomical models in the field of medicine. The proposed approach is based on the integration of modern computer graphics, medical image processing, and interactive modeling technologies [11].

The suggested methodology enables the development of a 3D anatomy simulator with a user interface in the Uzbek language, adapted to the local context of Uzbekistan. This approach plays a

significant role in improving the quality of medical education and enhancing doctors' practical skills through interactive learning methods.

In the future, it is advisable to expand such systems through VR/AR technologies and to implement automatic anatomical segmentation using artificial intelligence (AI) for more accurate and efficient modeling.

REFERENCES

1. Aldridge, R. D. (2022). *3D Visualization in Medical Education: Applications and Outcomes.* Journal of Medical Simulation.
2. Krupinski, E. A. (2020). *Medical Imaging and Visualization: Current Trends and Future Directions.* Academic Radiology.
3. 3D4Medical Ltd. (2023). *Complete Anatomy: Educational Software Manual.* Dublin.
4. Pieper, S., Lorensen, B., & Schroeder, W. (2020). *3D Slicer: A Platform for Biomedical Research and Visualization.* Insight Journal.
5. Unity Technologies. (2023). *Unity Engine for Medical Simulation and VR Training.* Whitepaper.
6. Mahmudova, M. A. (2025). *Scientific Basis for Developing a 3D Anatomy Simulator in the Context of Uzbekistan.* Tashkent.
7. Orthanc Team. (2024). *Open-source DICOM Server for Medical Imaging Integration.*
8. WHO eHealth Division. (2023). *Digital Transformation in Medical Education: Global Review.*
9. Blender Foundation. (2022). *Blender for Scientific Visualization Manual.* Amsterdam.
10. Muminov, A. B. (2024). *The Importance of VR/AR Technologies in Medical Education.* Journal of Scientific Innovations.

Saidov, I. R. (2025). *Integrative Possibilities of Medical Visualization and 3D Modeling.* TATU Scientific Bulletin.