🏰 3D Medieval Castle Design – My Approach

- by Nishit Sarvaiya

Geometry Creation

When building the 3D castle, I started by thinking modular — each architectural piece should be reusable, adjustable, and efficient.

- **Rectangular Base:**
  I used PlaneGeometry for the ground with dimensions 200×180. I rotated it by -Math.PI / 2 to lay it flat on the XZ plane. This serves as the foundational floor and nicely anchors the entire castle structure.
- **Towers:**
  For the main towers, I went with CylinderGeometry — a slightly tapered design with radiusTop smaller than radiusBottom for a stronger silhouette. I positioned two of them on opposite corners of the base.
- **Crenellations:**
  I placed small cube geometries evenly around the top using InstancedMesh and trigonometry to calculate circular placement. This approach keeps performance high without sacrificing visual detail.
- **Walls:**
  The walls connecting the towers are just BoxGeometry pieces. I positioned and rotated them appropriately to form the front, back, and sides of the castle.
- **Bushes:**
  I wanted some decorative greenery around the castle, so I created instanced bushes using SphereGeometry and a hand-curated array of position/ scale values. These really help ground the scene and add variety without adding too much overhead.

Material and Color Application

I designed the material system to be flexible and easy to toggle between performance-focused flat colors and fully detailed PBR textures.

- **Material Strategy:**
  For each major element — the floor, walls, towers, tower tops, crenellations, and bushes — I created both flat and textured materials. Flat materials are just basic MeshStandardMaterial with solid colors like blue, yellow, and green for a medieval but stylized look.
- **Texturing:**
  The textured versions use color maps, ARM (AO/Roughness/Metalness), normal maps, and optional displacement maps. These textures are loaded using my custom Textures loader class, which handles repeat wrapping and color space conversion.
- **Dynamic Switching:**
  I exposed a GUI toggle to switch between flat and textured modes. This is super useful for performance testing or running on lower-end devices.
- **Color Harmony:**
  I went with blue for the base and walls, yellow for the towers, and green for the bushes — which gives enough contrast to visually separate elements while maintaining a unified aesthetic thanks to consistent PBR shading.

Performance Optimization

Performance is something I always keep in mind, especially for real-time 3D scenes. Here's how I handled it:

- **Instancing:**
  The crenellations and bushes are instanced using InstancedMesh. This drastically reduces the number of draw calls and helps the scene run smoothly even with a lot of repeated geometry.
- **Geometry Reuse:**
  All major geometries are created once and reused wherever needed. This keeps memory usage lean and avoids redundant GPU uploads.
- **Efficient Textures:**
  I optimize texture usage by setting appropriate repeat values, converting to SRGBColorSpace, and skipping displacement maps if they don't exist — all through the Textures class.
- **Shadows:**
  I enable shadows selectively and keep shadow map resolution reasonable (512×512 for directional light). Only key objects cast or receive shadows, which avoids unnecessary GPU work.

User Experience

- **Orbit Controls:**
  I used OrbitControls and restricted the camera's polar angle to keep the view intuitive and prevent disorientation. Users can explore freely but won't accidentally flip the scene upside down.
- **Cinematic Camera Entry:**
  On load, I animate the camera using gsap from a wide view into a closer one. This subtle intro adds a polished feel and immediately sets the stage for the user.
- **Loading Feedback:**
  I implemented a custom loading screen with a progress bar and smooth exit animation. This helps set expectations and keeps users engaged even while assets load.
- **Environment:**
  I added a Sky object for realism and tuned it using turbidity, rayleigh, and mie scattering values. Combined with FogExp2, the scene gets a nice atmospheric depth that makes the castle feel like part of a world — not just a floating model.
- **Interactivity:**
  The GUI panel allows toggling textures, which adds a layer of control and makes testing or showcasing the model easier.

Final Thoughts

- Overall, I approached this castle as a clean, optimized, and modular 3D experience. The instancing, material system, and camera work together to make it feel lightweight but visually rich.
- If I were to expand this, I'd love to:
  - Add more modular components (like gates, interior buildings, or animated flags),
  - Introduce user interactivity (e.g., clicking towers to zoom in or get details),