ChatGPT

Holographic UI Pattern Library for Volumetric & Light-Field Displays

Introduction

Holographic user interfaces present information in **true 3D space**, leveraging depth and volume to create immersive experiences. Unlike flat screens, volumetric and light-field displays can provide all the natural depth cues (perspective, occlusion, shadows, motion parallax, stereopsis) that humans use for 3D perception [1]. This richness comes with unique design challenges. Interaction in 3D space must remain **intuitive, low-latency, and legible** to avoid confusion or discomfort. For a "Holographic Extractor" herb system – a conceptual interface that visualizes herb "signatures" (rings, petals, nodes) in 3D – we need patterns that support **spatial menus, depth cues, overlays, and mid-air gestures** in a solo or small-group setting. The following pattern library surveys proven volumetric/light-field UI techniques and proposes new patterns for herb signature visualization, along with interaction guidelines focused on usability, **latency**, and **legibility**.

Volumetric & Light-Field Interaction Overview

Volumetric Displays: These displays generate **true 3D images by illuminating points in space**, so viewers can walk around and see content from any angle [2]. Multiple people can view the same volumetric scene simultaneously without headsets [2]. A classic example is a spinning volumetric dome (swept-volume display) that projects a 3D image within a glass enclosure [3]. Volumetric UIs inherently solve the focus/accommodation problem of stereoscopic AR (no more convergence mismatch that causes eye strain [4]). However, they impose constraints: the imagery is typically enclosed in a transparent volume, so users **cannot literally touch the floating objects**. Early research noted that while all virtual objects are within arm's reach in a volumetric display, the protective enclosure means users must interact indirectly (no reaching in to grab objects as you might in VR) [5]. This led to input techniques like motion-tracked gloves or sensors to capture freehand gestures around the display [6]. For instance, Grossman et al. tracked users' fingers to directly manipulate 3D objects in a hemispherical volumetric display [6]. Such freehand interaction leverages the 360° viewing volume and natural depth perception [7]. One challenge is precision – mid-air gestures lack the tactile feedback of physical controls, so the UI must account for potential jitter or inaccuracy [8].

Light-Field & AR Displays: Light-field displays (including AR headsets with stereoscopic or multi-focal optics) create a holographic illusion for an individual user. Devices like Microsoft HoloLens or Magic Leap provide **"see-through" 3D visuals** superimposed on the real world. They often rely on **head/eye tracking and stereoscopic rendering**, which can introduce **latency** and focus issues if not carefully handled [9]. Designers must work within limited fields of view (FoV) and ensure content stays legible against unpredictable real backgrounds. Still, these systems support **intuitive inputs** (hand tracking, eye-gaze, voice) and allow users to walk around holograms. Many interaction patterns overlap with volumetric displays. For example, using freehand mid-air gestures and **gaze** is natural in AR – a user can point or pinch in the air to select a hologram at a distance [10]. Studies have combined **eye-gaze and hand gestures** for

AR menu selection, often using a **dwell time** on the target plus a hand "commit" gesture to confirm [11]. A key advantage of true light-field displays over basic stereo is reducing eye strain by matching focus cues, but this technology is emerging. For early prototyping, one might use projection-based volumetric setups (like a fog screen or Looking Glass display) that multiple people can see, then later transition the UI to head-worn AR. In both cases, the UI design should be **platform-agnostic**, focusing on spatial interaction patterns that can work on a tabletop projector or through an AR headset. The **user context** here is primarily a solo person performing a "herb ritual" with the holographic extractor, but we also consider a **small multi-user** scenario – for example, a collaborative installation where a few people observe or partake in the interface. This means certain UI elements should be **visible from different angles** or replicated for multiple viewers [12].

Below, we present a **pattern library of 3D UI design patterns** (with approximately 10–15 key patterns) relevant to holographic herb extraction. Each pattern includes a brief description and guidelines, with references to prior research or tech demos. We then provide general interaction guidelines focusing on **latency** (performance) and **legibility** (visual clarity), which are crucial across all patterns.

Pattern Library – Holographic UI Patterns

Spatial Menu Patterns

- **Radial Menus & Ring Menus:** Arrange menu options in a circle around a central point or the user's focus. Radial layouts are ideal in 3D because they accommodate **circular head or eye movements**, which users find more comfortable than linear scanning in AR [13]. Early HoloLens designers found that a radial menu controlled by head-gaze reduced neck strain compared to a flat menu [13]. In practice, items can be distributed evenly on a ring or sphere; the user highlights an item by facing it or pointing at it. A clever variant is the **rotating ring menu** – the menu items are arranged on a 3D ring (imagine a belt around a sphere) that the user can rotate so that the desired item comes to the front [14]. This pattern keeps all options accessible while only one subset is directly facing the user at a time. **Use Case:** In the herb extractor, a radial menu could appear around a selected herb or tool, with options like "Grind", "Heat", "Infuse" arranged circularly. The user gazes at an option and performs a **commit gesture** (e.g. an air-tap or pinch) to activate it [11]. Radial menus work well with **mid-air pointing** and gaze because the circular layout has no corners – as a result, it feels fluid to navigate. Research and practice show this design is effective for holographic interfaces [13], but designers should limit the number of menu slices to avoid crowding and use clear icons/labels for each segment.

- **Contextual Floating Menu:** Rather than a fixed HUD, menus can **follow the user's focus** or appear near the relevant object in space. For example, when the user selects an herb, a contextual menu might float just beside that herb's holographic representation. In a HoloLens case study, the team positioned their menu **about 1.5 meters from the user, to the left of the field of view or next to a selected object**, which made it easily reachable yet not overwhelmingly close [15] [16]. Keeping menus at ~1.5 m distance is a good rule of thumb for AR – at that depth, content stays **visible and comfortable** to focus on [16]. The menu was also set to always face the user (billboard) so that it's readable from the user's viewpoint [15]. **Use Case:** In our scenario, when a user grabs a specific herb jar in the hologram, a menu could pop up next to it (e.g. options to extract oils, view info, etc.), always orienting itself toward the user for legibility. **Caution:** Contextual menus should not obstruct the main scene. Designers found that if an object was very large or at the edge of the FoV, a menu

"stuck" to it could move out of view or take up too much screen space [17]. To mitigate this, ensure the menu has a sensible anchor (e.g. offset position so it's likely within view) and consider allowing the user to **reposition or pin the menu** manually if needed. Contextual menus shine in making the interface feel part of the scene, but they must be used judiciously to **not disturb the primary task**.

- **Hand or Tool-Anchored Menu:** In mixed reality, sometimes the user's own hand can serve as a platform for UI. For instance, on HoloLens 2, holding up one's palm can summon a menu attached to the hand (the menu might hover above the palm or be fixed to a controller device). This pattern keeps the menu **"always reachable"** since the user literally brings it into view when needed. In a volumetric display setup, an analogous pattern was to place controls on the **physical enclosure** – e.g. sliders or buttons on the rim of a volumetric dome [18]. For AR, we translate that idea into virtual controls on the user's hand or wrist device. **Use Case:** The herb extractor UI could allow the user to hold up their left hand to see a "toolbar" of common actions (a bit like a **diegetic** menu attached to a glove or bracelet). They could then tap selections with their right hand or via gaze. This pattern leverages proprioception – users know where their hands are – to locate the menu without searching the environment. It also naturally disappears when not in use (lowering the hand hides it). **Guidelines:** If using a hand menu, ensure the system reliably detects the palm-up gesture and place the UI at a readable angle (often slight tilt toward the user). Keep it small and light to avoid occluding too much of the real world. This technique is very convenient for solo use; however, it might not be visible to other users (so for multi-user, provide duplicate UI or an alternate menu control).

Depth Cues & Overlay Patterns

- **Depth Shadows and Anchors:** In a free-floating hologram, users can lose sense of how far objects are or how they relate to real surfaces. A useful pattern is to provide **visual depth cues** like **shadows, glows, or reference grids**. For example, a 3D object can cast a **drop shadow onto a virtual "ground" plane** beneath it, giving viewers a point of reference for its depth and height [19]. Research on volumetric UIs suggests using **projection shadows** or "magic lens" filters to help users interpret spatial relationships [19]. In an AR herb extractor, we might include a subtle circular shadow on the table under a floating herb graphic, so the user sees where it "would" contact the table. Another technique is an **anchoring line or pointer**: a line connecting a floating label or icon down to the object it refers to (this line can help trace what a floating UI element is attached to in depth). **Guidelines:** These cues must be subtle enough not to clutter the scene – often semi-transparent or only appearing when needed. Ensure shadows are **consistent with lighting** (e.g. if there's a virtual light source in the scene) so they feel natural. Depth grids (like a faint checkerboard floor) can also be toggled in a volumetric display to give a global reference frame. Overall, providing at least one stable reference (shadow, floor, wall) greatly improves depth perception and **legibility of spatial layouts** [20].

- **Occlusion and Collision Overlays:** Occlusion is a powerful cue – if one object overlaps another, we know which is front. In AR, designers should allow physical objects to occlude holograms when appropriate (and vice versa) to maintain correct depth ordering. For instance, if a real hand reaches behind a virtual object, ideally the hand is hidden partly by the hologram. Volumetric displays inherently depict correct occlusions since they're truly 3D. A specific overlay pattern is the **"X-ray" or Magic Lens**: a see-through panel or filter that the user can position to reveal obscured details [19]. For example, holding up a virtual lens could highlight the internal structure of an herb or show hidden data, effectively overlaying an alternate view. This is useful in complex scenes where you

might need to peel back layers. **Use Case:** The herb system might let the user hold a magnifying glass gesture to see microscopic details of an herb hologram – the area under the "lens" could show a magnified view or chemical composition overlay. **Guidelines:** Use occlusion deliberately – if important UI elements are at risk of being blocked by other objects, consider an outline or **halo effect** to keep them visible. A glowing **outline around an object** can indicate selection and also preserve the object's shape when it overlaps with similarly colored backgrounds. Many AR toolkits include such outline/highlight shaders for this reason. In summary, embrace occlusion for realism, but provide alternative representations (outlines, X-ray panels) to handle cases where occlusion would hide critical information.

- **Omni-Viewable Widgets:** For multi-user or 360° setups, design **3D UI elements that can be read from any direction**. Unlike a flat screen interface that assumes one front view, a volumetric or shared AR display might have viewers standing all around it. One pattern is to create **multifaceted widgets** – e.g. a floating menu cube with icons on all sides, or text that is duplicated on panels facing different angles [12] [21]. Researchers Balakrishnan et al. coined the term "omniviewable widgets" for interface elements that are accessible from any viewing angle [12]. For example, a 3D slider might be implemented as a ring around the object, so any user can grab a part of that ring from their side. Another trick is to have **dynamic billboarding or motion**: small UI elements that gently rotate or rock, so that at some point each side faces the user [12]. **Use Case:** In the herb extractor installation mode (small group use), key info like the herb's name or status could be shown on a **floating band or ring** around the herb hologram, repeated on all sides or spinning slowly. Thus, people standing in different spots all see a label right-side-up [12]. **Guidelines:** Keep text short and use symbols that are universally understood from orientation (numbers, icons). For text specifically, you might opt for **multiple text billboards** placed around a circle, rather than 3D text geometry (which could appear reversed from behind). Omnidirectional design ensures no participant misses information, supporting collaboration. It does come at a cost of possibly more visual clutter (since you're duplicating UI). A compromise is to show the info to the primary user and perhaps minimal indicators to others, depending on your user roles.

- **Translucent Panels for Legibility:** A persistent problem in AR is **text legibility** against varied backgrounds. A proven pattern is to render text and icons on **opaque or translucent backdrop panels**. By giving text a solid (or semi-solid) background, you ensure sufficient contrast regardless of the real-world scenery behind [22]. Developers at Dauntless XR recommend using at least ~70% opacity for text background panels in mixed reality [22]. For example, if the herb extractor displays a description of an herb's properties, the text could appear on a frosted-glass panel that hovers near the herb. This panel could be a soft gray with white or black text (high contrast). In one AR design iteration, a team chose a **plain light-grey backdrop with black text for their menu**, explicitly to ensure it was always readable in AR's unpredictable environments [23]. This improved legibility but made the UI visually heavy, so the opacity and color must be tuned carefully [24]. **Guidelines:** Use **sans-serif fonts** with sufficient weight (avoid very thin lines) and a font size large enough for the expected distance. A study suggests starting around **16–20 pt for body text and 24–32 pt for headings at ~1-2 m viewing distance** [25]. Also consider dynamic adjustments: the system can brighten or enlarge text in very bright environments, or users can have a setting to scale UI text. The herb system could even detect the background (via camera) and invert text color if needed (white text on dark panel vs black on light). A related pattern is **"hover to reveal text"**: in a radial menu, you might show only icons by default and reveal labels when the user focuses on an item [26]. This keeps clutter low but still provides legible text on demand. Overall, **legibility is king** – do not

hesitate to sacrifice some visual style (e.g., making a panel more opaque) if it dramatically improves readability in holographic view.

Gesture & Interaction Patterns

- **Gaze-and-Dwell Selection:** When direct touch isn't available, a combination of **eye or head gaze to aim, and a timed dwell or gesture to select** is a reliable pattern. Microsoft's HoloLens employs **gaze and commit**: the user looks at a target, then performs a secondary action like an "air tap" hand gesture, a voice command ("select"), or simply holding the gaze for a brief moment to activate [11]. This multimodal input increases accuracy because the gaze (head or eye tracking) provides precise pointing, while the separate "commit" reduces false activations. For instance, in our herb UI, the user could **focus their gaze on a specific herb node** in a network, and a circular progress indicator (hover ring) appears. If they keep looking for, say, 1 second, it triggers a selection (dwell activation), or they can say *"Open details"* to immediately activate with voice. The design guidelines for eye-gaze input warn against using a persistent visible cursor for eyes (since our eyes dart rapidly, a constant cursor can be distracting) [27]. Instead, the system should give **subtle feedback** when the user's gaze lingers on something, e.g. a highlight that **smoothly ramps up over 500–1000 ms** to confirm the target [28]. We should apply this in the herb system: when the user looks at a "signature" element (ring/petal/node), it could gently glow or a ring could start forming around it, confirming "we see what you're looking at." The user can then complete the action by a voice command or hand gesture. **Guidelines:** Calibrate dwell times to balance speed and accuracy – too fast and users may trigger things by accident when their gaze wanders; too slow and it becomes frustrating. Also, synchronize modalities: if using voice commands that take time (e.g. speaking a long spell or command), ensure the system knows to act on the object that was gazed at when the command began [29] (since the user's eyes might move before speech is done). Gaze+dwell is best for activating fairly large targets or for hands-free contexts; smaller or precise adjustments might need actual hand control.

- **Mid-Air Gesture Set (Direct Manipulation):** An immersive holographic UI should exploit the richness of **mid-air hand gestures**. Users can use their hands to pinch, grab, push, and swipe virtual elements, which feels natural and "Minority Report"-esque. Common gestures include: **air-tap or pinch** (bringing index and thumb together) to select or "click" a hologram; **grab** (closing the hand into a fist around an object) to pick up or move an object; **push** (poking forward with a finger or palm) to press a button or slider; **swipe** or wave to scroll or navigate; and **two-handed** gestures like stretch (hands apart) to scale an object. These should be leveraged in the herb extractor UI – e.g., the user might **grab and rotate** a floating herb model to inspect it, or perform a pinching motion on a ring UI to twist a dial for adjusting extraction intensity. Research shows that freehand interaction is very intuitive for 3D tasks; for example, one study let users manipulate a 3D map on a light-field display using pan, rotate, and zoom gestures captured by a Leap Motion sensor [30]. The participants could naturally treat the hologram like a real object, moving it within the volumetric display. **Guidelines:** Even though gestures are natural, they require clear feedback and some learning. Provide **visual cues** when a gesture is detected – e.g., highlight an object when the user's virtual hand collider intersects it, or show a transparent hand ghost image to hint at the correct gesture. Also consider **ergonomics**: mid-air interaction can cause fatigue if overused. Avoid requiring the user to hold their arms up for long periods ("gorilla arm" problem). Design the workflow so that frequent actions can be done with minimal motion or supported by rests. For instance, use short reach distances or allow the user to alternate between hands and voice to reduce strain. Precision selection can be challenging with big, wobbly arm movements, so for small targets (like a tiny node),

amplify the motion or offer an alternative selection method (maybe gaze assists the aim). Prior research notes that while mid-air gestures unleash expressive interaction, they might be **less precise than mouse or touch** [8], so the interface should tolerate some imprecision (e.g., use larger hitboxes, snapping, or confirmation steps for critical selections).

- **Volumetric Cursors and Raycasting:** In many 3D interfaces, a **virtual pointer or cursor** helps bridge the gap between the user's hand and distant objects. A pattern from volumetric displays is the **3D volumetric cursor** – for example, a small semi-transparent sphere that the user can move in 3D space as a cursor [31]. Because the cursor itself is a 3D object, it can convey its depth by occlusion and size, and being translucent ensures it doesn't hide the objects behind it [32]. Raycasting is another ubiquitous technique: the system casts an invisible (or visible) ray from the user's hand or controller forward, and where it intersects a hologram, that object gets highlighted. HoloLens 2, for instance, shows a **hand ray** extending from your hand for far interaction – you see a cursor where the ray hits a UI element. In our pattern library, we recommend providing a **cursor or ray for precision tasks** like selecting small herb nodes or menu items at a distance. **Use Case:** If the user performs a pointing gesture (index finger out), the system could project a light ray from that finger into the scene. The first herb or UI element it hits will be highlighted, and the user can then pinch to select it. This is analogous to "laser pointer" interactions in VR. **Guidelines:** Ensure the cursor or ray provides clear feedback when it **hits a target** (change color or form a ring on the target). If nothing is hit, perhaps let it fade out to avoid clutter. Also, align the ray with the user's perspective to minimize parallax issues – e.g. if using head-gaze, cast the ray from between the eyes forward; if using hand, some calibration might be needed so it feels aligned with the physical hand position. An advantage of volumetric cursors is that, unlike flat 2D cursors, they won't be confusingly hidden behind other objects – you can make them always somewhat visible (e.g., outline of the cursor still seen even if behind an object) because depth is unambiguous in the volumetric view [31]. This pattern improves **accuracy and reach**, complementing direct hand manipulation.

- **Responsive & Animated Feedback:** 3D UI elements should respond to user input with **clear, timely feedback** to overcome any latency. For example, a holographic button might depress or change color when "pressed" by a virtual hand, or a menu option might pulse when focused by gaze. Animations also help convey depth and transitions – e.g., when summoning a menu, it could **fade in and scale up** from a point, which draws the user's attention and indicates where it appeared. In the herb extractor UI, imagine the "herb signature" rings gently orbiting until the user interacts, at which point the specific ring slows down or faces the user as a cue that it's selected. **Latency** is critical here: the system should strive for a high frame rate and low input lag so that feedback feels instantaneous. The general goal in XR systems is to keep motion-to-photon latency under ~20ms to maintain the illusion that virtual objects are stable and reacting in real time. High lag not only frustrates the user but can cause disorientation or nausea [9]. Therefore, design your system to be lightweight, possibly sacrifice some graphical detail for performance, and use prediction techniques for head and hand movement if available. If the hardware or network (in multi-user cases) does introduce latency, mitigate it with **anticipatory cues**: e.g., if a voice command will take a second to process, show a listening indicator immediately so the user knows their input was received. **Guidelines:** Every interactive element should have at least two states (idle and hover/active) with distinct appearance, and possibly a pressed state. Use sounds or haptic buzz (if devices allow) to reinforce confirmation of actions. And always consider the **timing** – for instance, a highlight that blends in over 0.5 seconds (as recommended for eye-gaze targets [28]) smooths out the interaction

and masks tiny latency or jitter in gaze tracking. In short, responsive feedback and animation can make the interface feel alive and connected to the user's actions, which is vital in holographic UX.

Herb Signature Visualization Patterns

(These patterns propose how to represent and animate the unique "signature" of each herb in the holographic extractor, using rings, petals, and nodes as UI metaphors.)

- **Signature Rings (Halo & Orbit): Rings** can be used to encircle an herb's hologram, acting as both decorative indicators and functional UI elements. A ring around an object immediately draws attention to it (much like a halo or selection circle) and provides a reference for depth and scale. For the herb system, envision each herb's essence visualized as one or more luminous rings orbiting it. These rings could encode information – for example, a ring's **color or thickness** might show the potency of the herb, and its rotation speed might indicate an active process (a faster spinning ring could mean the herb is currently being "energized" or extracted). **Interactive Behavior:** The user could tap a ring to cycle through modes or grab and twist a ring to adjust a parameter (like tuning a dial). Because rings are symmetrical, orienting them for the user is straightforward – but if the user moves, the ring might always rotate to remain facing them if it contains readable markings. We can combine this with known patterns: a ring could effectively be a **circular menu**. For instance, small icons or glyphs could sit on the ring's circumference at cardinal points, and the user rotates the ring to bring a desired icon to the front (this echoes the ring menu pattern from earlier [14], but anchored to the herb object). **Animation:** When an herb is first activated, its ring might **materialize with a gentle pulse**, expanding from the herb outwards and then settling into orbit. If the herb is inactive, the ring could shrink or disappear. If multiple rings are present (like Saturn's rings), they can have a layered animation – maybe offset rotations or alternating orbits – to create a rich volumetric iconography. All these motions should be smooth and **low-latency** so they respond in real time to user input (e.g., if the user grabs a ring to stop it, it should halt promptly). **Guidelines:** Visually, rings should contrast with the environment (perhaps a soft glow or distinct color) and be semi-transparent so as not to totally block the view of the herb inside. They also act as a depth cue – a tilted ring gives a sense of 3D orientation. Leverage that by maybe tilting rings at slight angles to indicate layering. But ensure text or symbols on rings are always facing the user or otherwise readable (you might use **billboarded text that follows the ring's position**). The ring pattern provides a **unifying motif**: every herb might have a ring "signature" but perhaps different herbs have different styles (one has a single large ring, another has three concentric rings, indicating different properties). This consistency helps the user quickly identify interactive targets, improving overall legibility of the interface.

- **Signature Petals (Bloom Menu): Petals** suggest an organic, radial layout where UI elements protrude outward from a center, reminiscent of a flower. For a ritualistic herb interface, this is both thematically appropriate and functionally useful. A **petal menu** is essentially a radial menu given a petal-like shape – each option is a teardrop or leaf shape that extends from the center object. Petals can represent different aspects of an herb. For example, an herb's signature could include several petal panels around it, each showing an aspect: one petal might show its elemental affinity (with an icon or color), another shows its potency level, another could be a trigger to mix that herb with others, etc. These petals remain folded or semi-transparent until engaged. **Interactive Behavior:** When the user gazes at the herb or performs a "bloom" gesture (like opening their hand), the petals could **bloom open** – i.e., expand outward from the herb revealing their content (text or icons on the

petals). The user can then select a petal by touching it or dwelling on it. This functions as a contextual menu that is literally *blossoming* when needed. The concept is similar to the "Bloom" gesture in HoloLens (which launched the main menu) but here the bloom is a visual metaphor for the menu itself. Each petal being spatially distinct around the object helps avoid overlap and leverages depth (the petals can float at slightly different angles in 3D so they're all visible). **Animation:** Key to this pattern is a **smooth bloom animation** – petals might start almost flush with the object and then arc outwards (imagine petals opening). When an herb is deactivated or the user looks away, the petals could gently fold back in, so as not to clutter the space. The animation should take into account latency; it needs to respond as soon as the user indicates interest, with maybe a quick 200ms delay to confirm intention, then bloom over 0.5s. This provides a flourish while remaining responsive. **Guidelines:** Petals should be sized so that the user can easily target one at a time with their hand or gaze (space them out enough). Use distinct colors or icons for each petal to aid memory (e.g., the "info" petal is blue with an "i" symbol, the "mix" petal is green with an icon). Text on petals can be tricky since they are angled; a solution is to have the text **always face the user** even if the petal itself is angled – essentially each petal carries a small billboarded label. The petal pattern is an instance of a **spatial contextual menu** (options directly around the object) and ties nicely to the theme (herbs and flowers). It creates a visually pleasing effect and keeps the menu **tightly coupled to the object of interest**, which is good for context and reduces eye travel compared to a fixed UI far away.

- **Signature Nodes (3D Nodes and Links):** Many herbs or ingredients have relationships – perhaps certain herbs complement each other or are components of a larger formula. A **node-link visualization** can depict this in the holographic UI. **Nodes** are small spheres or glowing orbs that represent entities (herbs, effects, processes), and they can be connected by lines or beams of light to show relationships. For the herb extractor, you might have a **constellation of nodes** floating above the device: each node is an herb's "essence", and lines between nodes show interactions (maybe a recipe or a sequence in a ritual). This essentially creates a **volumetric graph** that the user can explore. Nodes can also represent states – for example, a node could light up when an herb is active, or multiple nodes can connect and merge into a new node symbolizing a mixture. **Interactive Behavior:** The user can reach out and touch a node to select that herb or element. They might even **drag nodes** to combine them – for instance, grabbing two nodes and pulling them together could initiate a mixing process (the UI then could animate the two nodes orbiting each other and forming a new linked node). Because nodes are point-like, a raycast selection works well (point and "click" a node). If nodes are numerous, implement techniques like **clustering or filtering** (perhaps use a magic lens: look through a filter to highlight certain types of nodes [19]). **Animation:** Nodes could exhibit lively behaviors: idle nodes might bob gently or emit particle effects (a faint aura) to indicate their type. When a node is selected or activated, it could *pulse* (grow and shrink slightly, or brighten). Links between nodes might appear dynamically – e.g., if the user selects an herb node, lines emanate from it to other compatible nodes, indicating what can be combined. These lines can be drawn with an animated stroke effect (like they light up and extend outward). The goal is to provide **spatial cues**: the lines themselves help the eye follow connections in 3D space, which might otherwise be hard to track. **Guidelines:** Keep the node design consistent (all nodes similar shape, different colors or icons for different categories). Use 3D positioning to avoid too much overlap – techniques like slight **depth separation** (put some nodes closer or farther) can reduce visual clutter, but then use connecting lines or drop shadows to maintain the sense of connection. Also, for legibility, consider a **focus+context** approach: emphasize the nodes that are currently relevant (bigger, brighter) and deemphasize others (smaller, translucent) rather than showing everything at

full intensity at once. This reduces cognitive load in a complex graph. From a performance standpoint, be mindful of too many dynamic elements – ensure the system can handle animating multiple nodes and lines at 60+ FPS (perhaps limit the total number shown or update them in groups if needed). Node graph patterns in AR have been used to visualize networks and can be very powerful, but they require careful **depth management** and user-friendly interaction (maybe allow the user to "step into" the graph or rotate the whole graph via gesture to see connections better). In our herb system, the node constellation gives a high-level "overview" mode, whereas rings and petals are more detail and control on individual items.

Interaction Guidelines (Latency, Legibility, and Usability)

Finally, we summarize key **guidelines** to ensure these patterns work together in a smooth, legible, and low-latency experience:

- **Optimize for Low Latency:** In holographic interfaces, system lag can break the illusion of reality and even cause discomfort. Aim for **responsive interactions** – ideally, under 20 milliseconds from input to displayed response. High frame rates (60 FPS or above) keep motion smooth and help holograms stay locked in place as the user moves. Prior experiences with head-tracked 3D have shown that lag and tracking inaccuracy lead to nausea and dizziness [9], so every millisecond matters. Use efficient rendering techniques and consider prediction (many AR systems predict head motion a few frames ahead to reduce apparent latency). If using networking (multi-user scenario), design the UI so that slight delays do not confuse – e.g., non-critical animations can be client-side only, and any shared state changes should be signposted with progress indicators if they take time. Always provide **immediate feedback** even for longer operations (for example, on a voice command, show a "listening" icon right away, or on a gesture hold, show a progress ring). This keeps the user confident that the system is reacting in real time.

- **Maintain Legibility and Contrast:** Because AR content is viewed against real-world backdrops and volumetric displays project in free space, always ensure **sufficient contrast** for text and important symbols. Use **high-contrast color schemes** (light text on dark background or vice-versa) [33] and avoid relying on color alone for critical info (consider color-blind friendly palettes and add shapes or labels). As mentioned, **opaque or translucent backgrounds** behind text are your friend [22]. This includes tooltips, labels, or menu text. Be mindful of ambient lighting – in very bright settings, holographic text can wash out. Dynamic adjustments like increasing brightness or switching to a bolder font in bright light can help (some AR devices have light sensors to facilitate this [34]). Also, choose fonts and sizes for readability: sans-serif, medium weight, and scale with distance. If the user can be at varying distances, you might implement **dynamic scaling** (text gets slightly bigger if the user steps farther away, to keep angular size consistent). For volumetric installations, consider the viewing volume – users at different angles or distances should all be able to read key info, which may mean duplicating text on multiple sides or using 3D text that faces each direction [21]. Always test the UI in diverse conditions: different room lighting, cluttered physical backgrounds, multiple simultaneous viewers, etc., to ensure legibility holds up [35] [36].

- **Use Depth Wisely (Avoid Overload):** Depth is a double-edged sword: it adds realism but can also confuse if too many elements are overlapping or if the user doesn't know where to focus. Follow the principle of **depth layering** – assign different "layers" of information to different depth ranges. For example, primary interactive elements could live in the 1–2 m range (comfortable focus), while

decorative background visuals can be farther, and any HUD indicators could be slightly closer at ~1 m so they always appear in front. Provide **depth cues** consistently: shadows, occlusion, perspective, motion parallax (by allowing the user or object to move) all reinforce understanding [1]. If an object is critical, ensure it's not floating ambiguously without context; give it a reference like a shadow on a surface or a connecting line. Avoid having too many semi-transparent overlays at different depths – this can cause "ghosting" or double vision in stereoscopic displays and is visually confusing. Instead, manage the UI complexity by **progressive disclosure**: show what is needed for the task at hand, and hide or minimize extraneous info (the approach where the menu hides during object manipulation is a good example [37]). Users can always bring back hidden UI, but when focusing on a task, fewer distractions improve depth perception and performance.

- **Support Multi-User Viewing:** If the experience involves more than one user (even just an observer), design the UI so it's not *only* optimized for a single viewpoint. This might mean choosing patterns like **omniviewable widgets** for shared data [12], or ensuring that one user's actions are signaled to others (for instance, if the primary user selects an herb, maybe a highlight appears on that herb for everyone). Volumetric displays naturally support multi-user since everyone sees the same floating image [2], but in AR headsets, each user has their own view – in a shared scenario, you'd network the experience so all users see consistent state (e.g., if herb A's ring turns red for one user, it turns red in all headsets). For co-located collaboration without headsets (like a group around a holographic table), consider **physical affordances**: people might point at things or gesture to each other. The system could potentially track laser pointers or physical wands if that helps multi-user interaction. Also, avoid UI that blocks another user's view – e.g., a panel that always faces User 1 will appear as a weird edge to User 2. One solution is to instantiate personal UI for each user (each sees their own version of menus) while keeping shared holograms in the center. In summary, decide on the interaction model (single active user vs. multi-user collaborative) and **ensure everyone can comfortably view and interact** with the holograms from their position.

- **Modalities & Redundancy:** Provide multiple input methods for flexibility and accessibility. Our design already integrates **hand gestures, gaze, and voice** – these should complement each other. For example, voice commands can act as shortcuts (saying "Combine herbs" might execute a series of steps that could also be done via manual interaction). In the HoloLens menu case, they allowed voice to **"double-map"** actions that were also on buttons, giving users a choice [38]. This is great for accessibility (someone with limited arm movement could speak commands) and efficiency (experts might prefer voice for speed). However, always have a non-voice alternative for noisy environments or when speech isn't possible, and likewise a non-gesture fallback if someone cannot use hands (perhaps a gaze-only mode). Redundancy also means giving **feedback in multiple channels**: visual highlight, auditory cue (like a subtle chime when an action is successful), and maybe haptic (if using a controller or wearable that can vibrate). This multi-sensory feedback helps overcome any single channel's limitations (e.g., if the room is loud, the user still sees the visual confirmation). It also reinforces the user's mental model that the system registered their input. For instance, upon successfully extracting an herb, the UI could play a brief success sound, flash the herb's node, and maybe show a text confirmation. This might seem excessive, but in a spatial environment where the user's attention could be anywhere, a bit of redundancy ensures the feedback is noticed.

- **User Comfort and Safety:** Holographic interfaces should be designed for **comfort over longer periods**. This means ergonomics: as mentioned, avoid prolonged arm-raised postures – design interactions to be quick or interspersed with rest. Encourage natural movement: allow the user to

walk around to inspect holograms from different angles (the UI should update smoothly as they do). Also, consider **personal space** – in AR, do not suddenly spawn a UI element very close to the user's face, as this can be startling or hard to focus on. The 1.5 m distance guideline is a safe minimum for most content [16]. Relatedly, avoid **visual overload**: too many flickering or moving elements can cause eye fatigue. Use motion purposefully (e.g., a slowly rotating object to indicate it's active, not just for flair) and allow the user to turn off or tune down certain effects if they find it overwhelming (maybe a "simple mode" without all the particle effects and animations). In a multi-user physical space, ensure that people aren't guided to walk somewhere dangerous because of the holograms – for instance, if an herb node is near a real wall, maybe keep it slightly away or warn the user via boundaries. On the positive side, volumetric and AR interfaces can be very engaging; just make sure this engagement doesn't come at the cost of physical strain or confusion.

- **Testing and Iteration:** Because this is cutting-edge UX, it's crucial to test these patterns with actual users in realistic settings. Many of the guidelines above (like contrast, distance, gesture comfort) benefit from empirical tuning. When implementing the herb extractor UI, test tasks like selecting a petal, combining nodes, reading text on a panel in various lighting, etc. Gather feedback on what feels intuitive versus awkward. You might discover, for example, that users prefer using gaze for small targets and hand for large ones, or that a certain gesture is frequently misrecognized – that's a sign to refine the gesture detection or choose a different gesture. Pay attention to **latency complaints** or misselection errors – they indicate technical or design adjustments needed (e.g., increase dwell time, enlarge hit zones). Also, watch for any signs of **fatigue or confusion** during longer sessions. The pattern library we've compiled is a starting point, but the optimal configuration for your specific context will come through iteration. The result should be a cohesive interface where the **spatial menus**, **depth cues**, and **gestural controls** all work in concert, allowing the user to carry out the herb extraction ritual as seamlessly as if it were a real physical task – with the added magic of holographic visualization.

Conclusion

Designing a holographic UI for a volumetric or light-field system requires balancing innovation with human-centered principles. We surveyed interaction techniques from volumetric displays (true 3D viewable imagery) and modern AR (head-tracked light-field) to inform a set of patterns tailored to the "Holographic Extractor" use case. The **pattern library** above enumerates solutions for spatial menus (radial menus, bloom petals), depth management (shadows, translucent panels, omni-view widgets), gesture control (gaze-and-commit, mid-air manipulations, volumetric cursors), and bespoke herb signature visuals (rings, petals, nodes with dynamic animation). Key references from research and practice underpin these patterns – from early volumetric UI experiments by Balakrishnan et al. [1] [2] to recent AR interface design lessons [13] [22] – ensuring our design is grounded in proven ideas. By following the **interaction guidelines** on latency (keep it fast), legibility (keep it clear), and user comfort, we can create an interface that is not only **immersive and magical** but also practical and user-friendly. In the end, a successful holographic UI allows the user to forget about the interface itself and focus on the task – whether it's extracting the essence of a virtual herb or collaborating in a mixed-reality ritual – with the interface invisibly supporting every intention and action in the three-dimensional canvas around them.

Sources: Key insights and patterns were informed by prior work in 3D user interfaces and mixed reality design [9] [2] [13] [12] [22] [11], including research on volumetric display interaction styles [6] [14], AR UX case studies [15] [39], and best-practice guides for spatial computing (Microsoft, etc.) [27] [28]. These

references (cited in-text) provide further reading on designing effective, user-friendly holographic experiences.

---

1 2 3 4 9 12 18 19 21 31 32 User Interfaces for Volumetric Displays
https://www.research.autodesk.com/app/uploads/2023/03/user-interfaces-for-volumetric.pdf_rec5GPLbHht12JW54.pdf

5 6 7 14 Microsoft Word - p219-grossman for cameraready pdf creation.doc
https://www.dgp.toronto.edu/~ravin/papers/uist2004_volumetric.pdf

8 30 The Good News, the Bad News, and the Ugly Truth: A Review on the 3D Interaction of Light Field Displays
https://www.mdpi.com/2414-4088/7/5/45

10 Gaze-Hand Alignment: Eye Gaze & Mid-Air Pointing for AR Menus
https://dl.acm.org/doi/10.1145/3530886

11 27 28 29 Eye-gaze and commit - Mixed Reality | Microsoft Learn
https://learn.microsoft.com/en-us/windows/mixed-reality/design/gaze-and-commit-eyes

13 15 16 17 23 24 26 37 38 39 Design an Immersive Menu for AR/VR: a Learning Journey between Ambition & Humility | by Alexia Buclet | Bootcamp | Medium
https://medium.com/design-bootcamp/design-an-immersive-menu-for-ar-vr-a-learning-journey-between-ambition-humility-361185a1438b

20 Effects of Volumetric Augmented Reality Displays on Human Depth ...
https://www.sciencedirect.com/org/science/article/pii/S1942390X19000016

22 25 33 34 35 36 How to Improve Text Readability in Mixed Reality Apps | Dauntless XR — Dauntless XR
https://www.dauntlessxr.com/news/how-to-optimize-text-readability-in-mixed-reality-3-steps-for-xr-developers