Unit 1 – Introduction & Graphics Systems

Q1. Computer graphics kya hai? Iske development, applications aur advantages explain karo.

Computer graphics ka matlab computer ki help se images, figures aur animations generate, modify aur display karna hota hai. Early stage me CRT displays aur simple line drawings use hote the, lekin aaj high-resolution raster displays, GPUs, CAD, gaming, animation studios sab computer graphics par depend karte hain. Development roughly teen phases me dekha ja sakta hai: batch plotting era, interactive graphics era (light pen, terminals), aur modern GUI/3D era jahan OpenGL, DirectX, VR/AR ka use hota hai. Applications me CAD/CAM, simulation, data visualization, GUI design, film VFX, gaming, education, medical imaging (CT/MRI visualization) include hote hain. Advantages: communication clear hoti hai (graphs/diagrams), design aur prototyping fast ho jata hai, what-if analysis possible hota hai, aur real-world experiments ki cost/safety issues kam ho jate hain. Is subject ka core idea ye hai ki kaise hardware (display devices, frame buffer, display processor) aur software (algorithms, standards) milकर screen par picture banate hain.

Q2. Raster scan aur random scan display me antar likho. Continual refresh display aur storage display ka concept samjhao.

Raster scan CRT me electron beam line-by-line top se bottom tak ja kar complete screen refresh karta hai; frame buffer me stored bits decide karte hain kaun-sa pixel glow karega. TV, monitors isi technique par based hote hain. Random scan (vector display) me beam sirf lines/curves ke path follow karta hai, isliye image wireframe jaise dikhti hai aur resolution geometrical accuracy par depend karti hai, fixed grid par nahi. Continual refresh displays me picture ko stable rakhne ke liye periodic refresh zaroori hota hai; agar refresh rok diya jaye to image fade ho jata hai (typical CRT). Storage display me phosphor ya special storage mechanism image ko long time tak hold karta hai, jise baar-baar refresh ki need nahi hoti; oscilloscope type devices iska example hain. Raster ka advantage filled shapes, color, animation; random ka advantage smooth lines aur kam memory requirement hai, lekin complex scenes me flicker aa sakta hai.

Q3. Color display techniques, frame buffer aur bit-blit (bit operations) ka concept explain karo.

Color display me har pixel ke liye color information store ki jati hai, jo RGB ya kisi aur color model ke components hoti hai. Monochrome display me 1 bit per pixel hota hai, jabki color raster displays me typically 8, 16, 24 ya 32 bits per pixel use hote hain; 24-bit me 16 million se zyada colors possible hote hain. Ye sab data frame buffer naam ke memory area me stored rehta hai, jahan har location ek pixel ko represent karta hai. Refresh hardware frame buffer ko sequentially read karke screen par show karta hai. Bit operations (bit-blit) me rectangular pixel blocks ko copy, move, mask, merge (AND, OR, XOR) kiya jata hai; isse window movement, cursor, sprites, icon drag-drop jaise operations efficiently perform hote hain. Color lookup table (CLUT) technique me pixel bits ek index represent karte hain jo palette me actual RGB value se map hota hai; isse limited bits me zyada logical colors handle kiye ja sakte hain.

Unit 2 – Drawing Algorithms & Scan Conversion

Q4. Scan conversion kya hota hai? Digital Differential Analyzer (DDA) aur Bresenham line drawing algorithm ke basic idea ko explain karo.

Scan conversion ka matlab continuous mathematical geometry (line, circle, polygon) ko discrete pixels me convert karna hai taa ki raster display par draw kiya ja sake. Line equation $y = mx + c$ ko use karke DDA algorithm incremental steps me next pixel compute karta hai: har step par x ya y me constant increment add ki jati hai aur doosri coordinate real value se round ki jati hai. DDA simple hai lekin floating-point operations aur rounding errors ke kaaran thoda slow aur less accurate ho sakta hai. Bresenham algorithm sirf integer arithmetic use karta hai; ye ek decision parameter maintain karta hai jo decide karta hai ki next pixel purely horizontal/vertical move se aayega ya diagonal move se. Is algorithm ka advantage hai: high speed, no floating point, constant time per pixel; isi wajah se hardware implementations me bhi use hota hai. Exam me line algorithm ke sath ek chota numerical example bhi poocha jata hai, jisme start-end points dekar pixels trace karne hote hain.

Q5. Midpoint circle algorithm ka principle likho. Circle ya ellipse generation ke liye midpoint method kyon useful hai?

Circle equation $x^2 + y^2 = r^2$ symmetric hota hai, isliye ek octant me pixels calculate karke baaki octants me mirror karke pura circle draw kiya ja sakta hai. Midpoint circle algorithm initial point $(0, r)$ se start karta hai aur har step par x ko increment karke decision parameter se decide karta hai ki y same rahega ya 1 se decrease hoga. Decision parameter circle ke true boundary ke midpoint ke sign par based hota hai; agar midpoint circle ke andar hai to ek pixel choice, bahar hai to doosri. Poore process me sirf integer addition/subtraction hoti hai, isliye algorithm fast aur hardware-friendly rehta hai. Ellipse ke liye bhi similar midpoint approach use hota hai, bas equation $x^2/a^2 + y^2/b^2 = 1$ ke basis par region-wise decision parameters define kiye jate hain. Midpoint methods aliasing kam karne me help kar sakte hain kyunki ye curve ke as-close-as-possible pixels choose karte hain, lekin pure anti-aliasing ke liye aur techniques (intensity control, supersampling) bhi use hoti hain.

Q6. Polygon filling techniques aur anti-aliasing ka concept explain karo.

Polygon filling ka goal hai boundary ke andar ke sare pixels ko choose karke color fill karna. Common methods: scan line polygon fill (har horizontal scan line ke liye edge intersection nikal kar even-odd rule se interior pixels fill karna) aur seed fill (boundary fill / flood fill) jahan ek interior seed pixel se recursively neighbors fill kiye jate hain. Scan line method large complex polygons ke liye efficient hai kyunki ye zyada memory nahi leta aur har line par linear operations karta hai; edge table aur active edge list use hoti hai. Aliasing jagged stair-step effect hai jo discrete pixels se lines/curves draw karne par dikhta hai. Anti-aliasing me intensity smooth karne ke liye techniques use hoti hain jaise supersampling (high-resolution me draw karke downsample karna), area sampling, ya pixel ke coverage fraction ke according gray level/alpha adjust karna. Isse lines aur text smoother dikhte hain, specially slanted lines par.

Unit 3 – 2D Transformations, Viewing & Clipping

Q7. 2D transformations (translation, rotation, scaling) ko homogeneous coordinates aur matrix representation ke sath explain karo.

2D geometry me basic transformations: translation (object ko move karna), rotation (origin ya kisi point ke around ghumana) aur scaling (size change karna) hoti hain. Normal 2D coordinates (x, y) ko homogeneous form $(x, y, 1)$ me likha jata hai, jisse sab transformations ko ek 3×3 matrix ke roop me represent kiya ja sakta hai. Translation matrix me last column me t_x, t_y terms aate hain, rotation matrix me cosθ, sinθ entries, aur scaling matrix me diagonal par S_x, S_y values hoti hain. Composite transformation ka matlab multiple transformations ko ek single matrix product me combine karna; actual effect order par depend karta hai, kyunki matrix multiplication non-commutative hota hai. For example, pehle scale phir translate karna aur pehle translate phir scale karna alag result dega. Screen pipeline me pehle modeling transforms, phir viewing, phir window-viewport mapping apply ki jati hai.

Q8. Window aur viewport kya hote hain? Window-to-viewport transformation ki need aur basic formula explain karo.

World coordinate system me programmer objects natural units (meters, logical units) me define karta hai; isme se jo region display karna hai usse window kehte hain. Viewport display device ke normalized ya device coordinates ka wo rectangular region hai jahan window ka content map hota hai. Window-viewport transformation scaling + translation ka combination hai jo window $\left(X_w^{min}, Y_w^{min}\right)$ – (X_w^{max}, Y_w^{max}) ko viewport $\left(X_v^{min}, Y_v^{min}\right)$ – (X_v^{max}, Y_v^{max}) me map karta hai. Scaling factors roughly: $S_x = (X_v^{max} - X_v^{min})/(X_w^{max} - X_w^{min})$ and similar for S_y, phir translation apply hota hai. Isse zooming, panning, multi-window displays possible hote hain, aur same scene different resolution devices par easily show kiya ja sakta hai. Clipping algorithms (jaise Cohen-Sutherland, Liang-Barsky) window ke bahar ke parts remove karke sirf visible segment ko pass karte hain.

Q9. Line clipping ke liye Cohen–Sutherland algorithm ka basic idea likho.

Cohen–Sutherland algorithm rectangular clipping window ke liye fast line clipping provide karta hai. Har endpoint ko 4-bit region code assign kiya jata hai jo batata hai ki point window ke left/right/top/bottom side me hai ya andar. Agar dono endpoints ka code 0000 hai to pura line accept; agar unka bitwise AND non-zero ho to pura line reject, kyunki dono ek hi bahar wale region share kar rahe hain. Remaining cases me line partially intersect karta hai, to algorithm boundary ke sath intersection point compute karta hai, endpoint ko replace karta hai aur codes re-calculate kar ke process repeat karta hai. Is incremental test-and-clip ki wajah se sirf genuinely ambiguous cases me hi exact intersection nikalna padta hai, jisse performance improve hoti hai. Is algorithm ko exam me theory + chota sa numerical example ke sath pucha jata hai, jahan given endpoints ke liye codes aur clipped line find karni hoti hai.

Unit 4 – 3D Concepts, Representation & Viewing

Q10. 3D transformation (translation, rotation, scaling) aur 3D viewing pipeline ka overview likho.

3D me points (x, y, z) homogeneous form $(x, y, z, 1)$ me represent kiye jate hain aur 4×4 matrices se transform hote hain. Translation matrix me last column par t_x, t_y, t_z hote hain; scaling me diagonal entries S_x, S_y, S_z hoti hain. Rotation x, y, z axes ke around alag-alag rotation matrices se hoti hai, jahan cosθ, sinθ terms rows-columns me arrange kiye jate hain. 3D viewing pipeline: model coordinates → world coordinates (modeling transforms) → eye/view coordinates (viewing transform – camera position/orientation) → projection (parallel ya perspective) → clipping → viewport mapping. Perspective projection me distant objects chhote dikhte hain, vanishing point create hota hai, jabki parallel projection me size distance se change nahi hota, jo engineering drawings ke liye useful hai. Hidden surface removal algorithms (z-buffer, painter's algorithm, scanline) visible faces select karte hain, takki final image correct depth cue de.

Q11. 3D object representation ke common methods (polygon mesh, quadric surfaces, spline/Bezier curves & surfaces) explain karo.

3D objects ko approximate ya exact mathematical models se represent kiya jata hai. Polygon mesh representation me object surface ko small flat polygons (mostly triangles/quads) ke network ke roop me store kiya jata hai; gaming aur real-time rendering me ye sabse common format hai. Quadric surfaces sphere, cylinder, cone jaise objects ko quadratic equations se define karte hain; ye mathematically smooth aur compact representation dete hain. Spline aur Bezier curves/surfaces control points ki set se smooth shapes define karte hain; curve in general control polygon ko approximate karti hai, sirf endpoints pass karti hai. Bezier surfaces 2D parameter domain me control points grid se define hote hain, jo car bodies, aircraft wings jaise free-form surfaces design me kaam aate hain. These higher-level representations design aur editing friendly hote hain, jabki rasterization ke time par inka polygonal approximation generate kiya jata hai.

Q12. Hidden surface removal ke z-buffer aur painter's algorithm ka basic concept likho.

Hidden surface removal ka goal hai observer ke view se sirf visible faces/pixels show karna. Z-buffer algorithm har pixel ke liye ek depth value (z) store karta hai; rendering ke time har new fragment ka z compare karke, agar woh camera ke zyada paas hai to color aur depth dono update kiye jate hain. Memory cost high hai lekin implementation simple hai, isi liye modern GPUs is approach ko use karte hain. Painter's algorithm me objects ko unke depth order me back-to-front sort kar ke draw kiya jata hai; peeche wale pehle paint, aage wale baad me, jisse final image me sirf front surfaces dikhen. Problem tab aata hai jab objects depth me intersect karte hain; is case me splitting ya special handling ki need hoti hai. Scanline algorithm ek horizontal line ke along visible segments compute karta hai aur active edge list + depth comparison se per-scanline visibility resolve karta hai.

Unit 5 – Animation, Tweaking, Morphing, GKS & Multimedia

Q13. Computer animation kya hai? Keyframe, tweening/tweeking aur morphing ka concept explain karo.

Computer animation me static images ko time ke according sequence me change karke motion ka illusion create kiya jata hai. Keyframe technique me animator sirf important poses ya frames define karta hai; intermediate frames automatically generate kiye jate hain, jise tweening/tweaking kaha jata hai. Tweening mathematically position, rotation, scale, color ya doosre properties interpolate karke smooth motion banata hai. Morphing ek special effect hai jahan ek image/shape gradually doosre me transform hota hai; isme dono objects ke feature points map karke geometry + texture interpolation ki jati hai. Animation pipeline me modeling, rigging (skeleton), keyframing, interpolation, rendering aur compositing steps include hote hain. Real-time animation (games) me physics-based motion, inverse kinematics aur GPU-accelerated skinning ka use hota hai; pre-rendered films me high-quality global illumination aur complex simulations use ki jati hain.

Q14. GKS (Graphical Kernel System) kya hai? Iske primitive functions aur graphics standardization ki need explain karo.

GKS ek early international standard graphics API hai jo 2D graphics ke liye device-independent interface provide karta hai. Iska objective ye hai ki application program ko hardware details se independent banaya ja sake, taki same code different terminals, plotters, printers par minimal change ke sath chale. GKS primitive functions me basic output primitives (polyline, polymarker, text, fill area), attribute settings (line type, color, text font), input primitives (locator, pick, choice, valuator) aur workstation control operations include hote hain. Standardization isliye zaruri hai kyunki agar har vendor apna alag graphics API de to portability aur maintenance huge problem ban jati. Later standards jaise PHIGS, OpenGL, Direct3D ne 3D aur advanced features cover kiye, lekin unka base idea bhi yahi hai ki graphics functionality layered, device independent aur well-defined ho.

Q15. Multimedia application kya hoti hai? Computer graphics ka multimedia me kya role hai?

Multimedia application me text, images, audio, video, graphics, animation aur interactivity combined form me use ki jati hai, jaise e-learning packages, games, presentations, websites, kiosks. Computer graphics isme core role play karta hai kyunki UI layout, icons, charts, diagrams, 2D/3D models, transitions aur visual effects sab graphics algorithms par based hote hain. Authoring tools (Flash/Animate types, presentation software, game engines) internally raster graphics, vector graphics, transformations, clipping, alpha blending, sprite animation use karte hain. Multimedia pipeline me capturing (scanner, camera, mic), editing (image/audio/video editors), compression (JPEG, MPEG), storage and delivery (CD, web, streaming) steps include hote hain. Effective multimedia design ke liye human-computer interaction, color theory, typography aur layout principles ke sath-sath graphics hardware capabilities ka bhi dhyan rakhna padta hai.