TEHNOLOGII AVANSATE PENTRU RECONSTITUIREA ŞI ACCESUL LA OBIECTIVE DIN PATRIMONIUL CULTURAL NAŢIONAL

Reconstituirea 3D a imobilelor distruse după fotografii vechi

Laura Ciocoiu *ciocoiu@ici.ro*

Dragoş Nicolau *dragos@ici.ro*

Dragoş Barbu *dbarbu@ici.ro*

Dragoş Smada *smada@ici.ro*

Valentin Radut *vradut@ici.ro*

Institutul Naţional de Cercetare - Dezvoltare în Informatică - ICI, Bucureşti

Rezumat: Lucrarea de faţă prezintă o componentă software, destinată reconstituirii 3D a imobilelor distruse, reconstituire realizată după fotografii vechi, ca o dezvoltare a instrumentului de reconstituire 2D, care redă atmosfera sfârşitului de secol XIX şi evoluţia aşezării urbane în diferite perioade.
În cadrul acestei lucrări, se încearcă o reconstituire 3D a clădirilor dispărute în momentul de faţă. Reconstituirea se va face după informaţiile oferite de către fotografiile existente în arhivă. În plus, vom simula accesul în clădire şi vom oferi iluzia vizitării interiorului încăperilor (mobilier, personaje etc.).

Cuvinte cheie: algoritmi de simulare, reconstrucţie digitală, descompunere ierarhică.

1. Introducere

Societatea modernă este o Societate Informaţională, caracterizată de fenomenul de diseminare a informaţiei din diferite domenii sub forma electronică, diseminare facilitată de posibilităţile de comunicare prin intermediul reţelelor de calculatoare. Acestea au condus la o adevărată revoluţie a conceptelor de structurare şi regăsire a informaţiei.

Principalele modalităţi de diseminare a informaţiei sunt: reţeaua Internet, prin intermediul modelului de tip *hypertext* şi ***CD-I (Compact Disc Interactiv)*** utilizând tehnicile ***multimedia,*** care se bazează pe combinarea diferitelor moduri de reprezentare a informaţiei de tip imagine, sunet, film. Astfel, percepţia informaţiei este îmbunătăţită prin asociativitate, prin structurarea acesteia şi prin exploatarea ei în mod nelinear, dirijat interactiv de utilizator, potrivit cerinţelor de informare, facilitând astfel regăsirea şi înţelegerea conceptelor.

În urma realizării muzeului virtual (accesibil în Internet la adresa **http://museum.ici.ro** şi pe **CD multimedia** *„Muzee din România"*), ce are drept conţinut **Muzeele din România**, s-a observat interesul manifestat pentru cunoaşterea şi reconstituirea trecutului. Informaţiile despre evoluţia aşezărilor urbane prezintă interes în educarea tinerei generaţii, dar nu numai.

Trebuie subliniat faptul că rolul principal l-au avut muzeografii care au furnizat informaţiile în calitate de colaboratori şi consultanţi, informaţii necesare reconstituirii evoluţiei urbanistice. Girul acestor specialişti este esenţial pentru recunoaşterea autenticităţii informaţiei.

În străinătate, s-au dezvoltat proiecte ce folosesc reprezentări avansate şi metode de vizualizare în dezvoltarea de software care converteşte datele în modele 3D. Noua tehnologie dezvoltă aplicaţii avansate în astfel de domenii, cum sunt grafica computerizată, animaţie şi efecte speciale. Aceste cercetări au fost expuse la conferinţa SIGGRAPH 2002, despre grafică computerizată şi tehnologie interactivă.

2. Obiective

Lucrarea de faţă **Reconstituire 3D** îşi propune:

- să constituie un instrument de informare a tinerei generaţii;
- aducă la cunoştinţa românilor şi străinătăţii farmecul şi culoarea specifică acestor zone;
- să prezinte străinătăţii informaţii corecte despre patrimoniul cultural al României;
- să trezească curiozitatea pentru evenimentul cultural, pentru România şi să faciliteze cooperări cu muzee şi centre culturale similare.

Pentru realizarea acestor obiective, se utilizează instrumente specifice de organizare şi regăsire a informaţiei prin tehnici de reconstituire de imagini 3D după fotografii vechi. Scopul este de a prezenta informaţia sub diferite forme (imagine 3D având la bază fotografii vechi şi sunet) şi de a realiza legături cu alte domenii şi servere de informaţii din ţară şi străinătate.

Produsele software necesare realizării reconstituirii 3D sunt: **Adobe Photoshop** - destinat digitizării şi prelucrării de imagini; software-ul realizat în **Visual C++**, care permite generarea **structurii ierarhice a** obiectelor (suprafeţe, elemente constructive / decorative etc.) ce definesc clădirea; software-ul specializat în gestionarea bazelor de date (My**SQL** ca server de baze de date); limbajul pentru gestionarea evenimentelor necesare parcurgerii/virtualizării ierarhiei de obiecte (**VRML**).

3. Algoritmi de reconstituire 3D a unei clădiri

Realizarea reconstituirii 3D, constă în:

- analiza şi procesarea geometrică a fotografiei;
- descompunerea manuală a imaginii în suprafeţe;
- generarea meşelor pentru simularea spaţiului 3D;
- virtualizarea spaţiului pentru generarea suprafeţelor ascunse, a interioarelor;
- crearea bazei de date arborescente, care conţine clasele rezultate din descompunerea ierarhică a unei clădiri. Fiecare element este definit prin atributele sale (tip, coordonate, legături între obiecte/elemente). Baza de date referitoare la reconstituirea unei clădiri conţine date extrase din sursele recomandate de specialişti şi completate cu informaţii specifice (imagini, hărţi, explicaţii contextuale);
- animarea reconstituirii, ceea ce dă posibilitatea vizualizării faţadelor nevizibile.

Prima fază a reconstituirii constă în **analiza grafică a clădirii** şi **descompunerea manuală** a imaginii în suprafeţe prin trasarea de contururi de culori diferite, în funcţie de tipul elementului (faţada frontală, lateral dreapta/stânga, principală [1], [2].

Descompunerea clădirii va consta în trasarea conturului:

- roşu - faţadele,
- albastru - elemente constructive de pe faţadă şi faţetele acoperişului,
- verde – elementele decorative.

Pentru realizarea descompunerii clădirii, s-a utilizat **Adobe Photoshop**. Pentru marcarea atributelor elementelor (coordonate x-y, poziţionare faţă de o suprafaţă de referinţă, culoare), s-a folosit **Visual C++**.

Pentru generarea **structurii ierarhice a** „obiectelor" ce definesc o clădire s-a folosit **Visual C++**; gestionarea bazei de date, care conţine descrierea elementelor componente şi a legăturilor dintre ele s-a realizat utilizând My**SQL**.

3.1. Generarea automată a meşurilor

Între 2D şi 3D, se află generarea suprafeţei meşului. Procesul de creare a meşului poate fi definit ca fiind procesul de împărţire a unui domeniu fizic în subdomenii mai mici (elemente) pentru a facilita găsirea unei soluţii numerice a unei ecuaţii diferenţiale parţiale. Meşing-ul este o parte integrată în procesul de analiză. Meşul influenţează acurateţea, convergenţa şi viteza soluţiilor. Timpul în care se creează modelul meşului este cea mai semnificativă parte a timpului de creare a unei soluţii; cu cât se creează mai repede meşul şi cu cât este mai automatizat instrumentul de meşing, cu atât va fi mai bună şi soluţia [2],[4],[5].

În timp ce acest proces poate fi folosit într-o multitudine de aplicaţii, cea mai răspândită este **metoda elementelor finite**. Domeniile suprafeţei pot fi divizate în forme de triunghiuri sau patrulatere, în timp ce volumele pot fi divizate în forme de tetraedru (piramidă triunghiulară) sau hexaedru (poliedru cu 6 feţe, cum este cubul). Forma şi distribuţia elementelor este definită ideal de algoritmi automaţi de creare a meşurilor.

3.2. Virtualizarea Realităţii şi Vizualizarea

Tehnicile de redare a imaginii permit integrarea reprezentărilor virtuale a lumii reale în aceste noi concepte media. În multe cazuri, căutarea realismului în reprezentările virtuale nu este doar de dorit, dar este şi un atribut esenţial. Anumite aplicaţii necesită un nivel suficient de asemănător cu cel original. Asemenea situaţii apar de multe ori în documentările despre patrimoniu sau în simulări şi învăţare şi în aplicaţiile industriale. Tehnicile de redare imagine au un rol esenţial în atingerea acestui obiectiv, prin folosirea datelor fizice ca o bază pentru acest proces [15],[16].

Lucrarea de faţă este orientată către căutarea şi dezvoltarea de astfel de tehnici. Îmbunătăţiri semnificative sunt necesare în inspectarea şi procesarea imaginii, în special către creşterea substanţială a flexibilităţii şi a automatizării, pentru a permite extinderea utilizării acestei tehnologii în general.

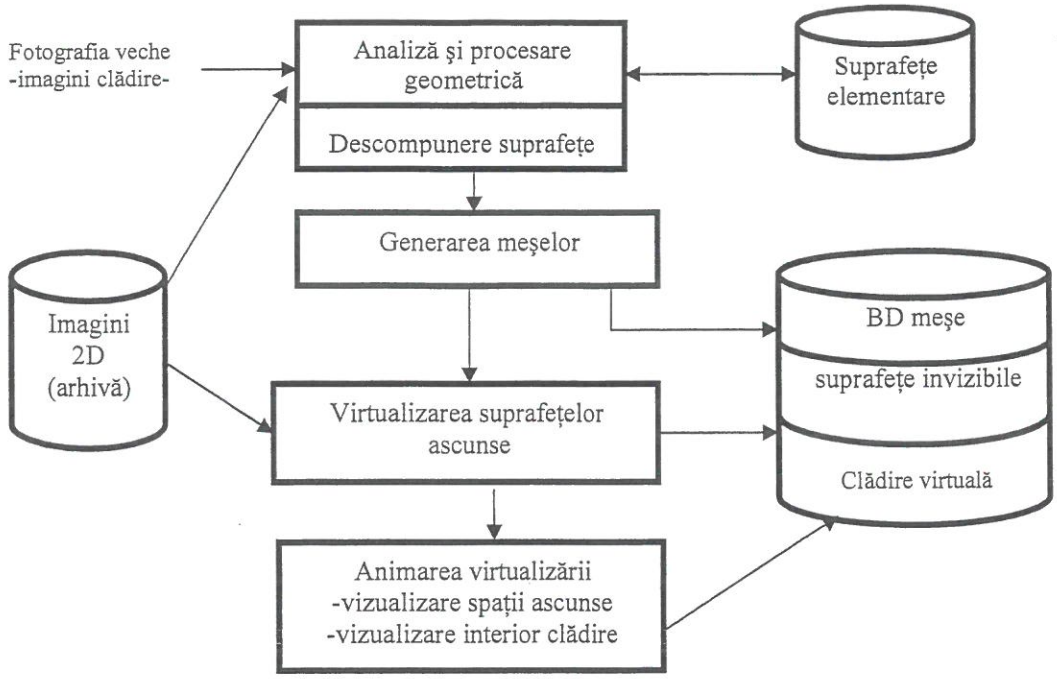


Figura 1 Schema de realizare a Reconstituirii 3D

După ce s-a realizat generarea meşurilor, se poate virtualiza şi anima reconstituirea.

Reconstituire 3D din Fotografii Vechi

În acest proiect, se urmăreşte dezvoltarea de software pentru reprezentări avansate şi elaborarea de metode de vizualizare pentru convertirea fotografiilor vechi în modele 3D. Noua tehnologie dezvoltă aplicaţii avansate în astfel de domenii cum sunt grafică computerizată, animaţie şi efecte speciale. [17].[20],[21]

Metodele de reconstrucţie 3D tradiţionale, bazate pe imagini, folosesc imagini multiple pentru a extrage geometria 3D. Totuşi, nu întotdeauna este posibil să obţinem astfel de imagini, cum ar fi cele de reconstrucţie a structurilor distruse folosind fotografiile existente sau picturile cu o perspectivă adecvată.

Metode pentru reconstrucţia 3D dintr-o singură imagine există. Ele folosesc blocuri ale formelor cunoscute pentru reconstrucţia dintr-o singură imagine. Acestea se mai folosesc şi pentru reconstrucţia de la un model de cameră fotografică folosind linii ascunse şi puncte ascunse. Abordarea nu are nevoie de modele ale obiectelor sau de parametrii acestora, nu are nevoie nici de linii sau puncte ascunse, care uneori nu sunt disponibile sau foarte greu de extras. Se folosesc diferite tipuri de constrângeri: constrângerii de tip punct/coordonată, constrângeri de suprafeţe şi constrângeri topologice.

Automatizarea modelării bazată pe imagini

Automatizarea modelării este bazată pe imagini care cercetează modul cum noile tehnici pot oferi o utilizare pe scară largă a automatizării procesului de modelare 3D.

Modelarea tridimensională din imagini, când este făcută numai de câte un om, poate dura foarte mult şi fiind nerealizabilă pentru proiecte de dimensiuni mari. Pe de altă parte, metodele complet automate nu pot obţine sau nu pot fi destul de precise (exacte) pentru multe aplicaţii cum ar fi documentarea „moştenirii". Modelarea tridimensională din imagini are nevoie de extragerea trăsăturilor (caracteristicilor), cum ar fi colţurile, şi apariţia lor în imaginile multiple. Totuşi, în situaţiile practice, aceste trăsături nu sunt întotdeauna disponibile, uneori nu apar în imaginile singulare, datorită ocluziei sau absenţei texturii suprafeţei. [6],[8],[10]

Pentru automatizarea modelării bazată pe imagini se are în vedere:

- folosirea tehnicilor automate cât şi cele interactive, fiecare unde se aplică mai bine, pentru obţinerea modelului obiectelor complexe cât mai precis şi complet;
- concentrarea către automatizarea construcţiei suprafeţelor nemarcate cum sunt coloanele, arcadele dintr-un număr minim de indicii disponibile;
- extragerea coţurilor obstrucţionate sau invizibile din suprafeţele şi liniile existente;
- exportarea coordonatelor 3D şi a modelelor în diferite formate 3D;

- modelarea 3D din două sau mai multe fotografii;
- modelarea 3D dintr-o singură fotografie;
- extragerea informațiilor în diferite formate 3D: prototipuri rapide STL şi Wavefront OBJ (cu MTL), în plus față de VRML şi CAD DXF;
- o gestionare îmbunătățită al mulțimilor şi atribuirea diferitelor tipuri de suprafețe fiecărei mulțimi;
- trasarea meşurilor triunghiulare pe orice imagine; toate mulțimile triunghiulare sau fiecare mulțime triunghiulară individuală poate fi trasată pe o imagine;
- vizualizarea punctelor de ştergere;
- introducerea directă a mulțimilor de coordinate 3D;
- măsurarea dimensiunilor dintre puncte;
- combinarea modelelor create independent, din diferite mulțimi de imagini într-un singur model: de exemplu, un model creat din imagini apropiate ale unei fațade a unei clădiri pot fi combinate cu modelul creat de imaginile depărtate ale aceleaşi clădiri.

După crearea unui model dintr-o mulțime de imagini, imagini adiționale, obținute cu scopul de îmbunătățire a texturii, pot fi înscrise cu acest model.

Modelul structurii de legare poate fi vizualizat şi editat folosind un editor 3D. Acest lucru este folositor la înlăturarea triunghiurilor nedorite. Modelul poate fi editat, de această dată adăugând sau înlăturând triunghiuri, din proiectul structurii de legare al oricărei imagini:

- calcularea coordonatelor 3D ale trăsăturilor extrase, din două sau mai multe imagini;
- adaptarea datelor la modelele geometrice standard: când punctele 3D cad pe o suprafață care este plană, sferă, cilindru;
- modelare geometrică şi trasarea texturii pentru vizualizarea datelor.
- combinarea modelelor create independent, din diferite mulțimi de imagini, într-un singur model; de exemplu, imaginile apropiate ale unei fațade a unei clădiri dintr-un model creat, din pot fi combinate cu modelul creat de imaginile depărtate ale aceleaşi clădiri, sau un model a unei secțiuni a unei camere poate fi combinat cu alt model al altei secțiuni a camerei.

Modelarea este automată pentru vârfurile îmbinate. [9],[12],[14]

- trasarea texturii: distorsiunea perspectivă este înlăturată din texturi.

Operațiile de procesare a imaginii constau în:

- limitarea automată (folosită în extragerea vârfurilor şi câte operații referitoare la câteva muchii);
- reducerea distorsiunii;
- ajustare şi evidențiere;
- detectarea muchiilor.

Trasarea texturii din modelele 3D este o ramură a cercetării în domeniul dezvoltării şi aplicării tehnicilor imagistice pentru reprezentări virtuale pentru a crea o experiență virtuală extrem de realistică. Aceasta se ocupă cu vizualizarea real-time şi manipularea interactivă a suprafețelor extrem de texturate. În acest proiect, trasarea texturii se aplică la două concepte ale reprezentărilor virtuale: filme de vizitare foto-realistice şi modele 3D, care permit utilizatorului vizualizarea interactivă şi manipularea. [7]

3.3. Animarea reconstituirii 3D

După realizarea reconstituirii propriu zise, se urmăreşte animarea acesteia în scopul vizualizării 3D a imaginii care să includă vizualizarea fațadelor nevăzute în imaginea 2D precum şi interiorul 3D al clădirii, sintetizat din descrierile existente în documentele de epocă. Acest lucru se va face folosind tehnologia OpenGL, care se axează pe utilizarea tipurilor de date (structură şi funcții) cu ajutorul cărora se prelucrează imagini şi se generează efecte vizuale remarcabile.

Produsul software OpenGL este destinat graficii cu scopul de a memora obiecte bi- şi tridimensionale într-un spațiu de tip cadru. Aceste obiecte sunt reprezentate ca secvențe de aşa numite „vertexuri" (blocuri de informație ce definesc obiecte geometrice) sau ca secvențe de pixeli (folosiți pentru definirea imaginii). Tehnologia OpenGL utilizează mai mulți paşi logici pentru a converti toată informația grafică

în pixeli, deci pentru a memora în spațiul cadru imaginea dorită, deci „produsul finit". În continuare, ne vom ocupa de prezentarea conceptelor fundamentale ale acestei tehnologii. [8]

OpenGL desenează primitive, adică puncte, segmente de dreaptă şi poligoane, supuse unor modalități multiple de selectare. Se pot crea mai multe moduri de selectare, independente unele de altele, dar care pot interacționa reciproc pentru a determina în ce fel va fi umplut finalmente spațiul.

Primitivele sunt definite printr-un grup de vertexuri. Fiecare din aceste vertexuri defineşte un punct terminal al unui segment sau un colț de poligon, unde se întâlnesc două muchii. Unui vertex i se asociază o cantitate de date (coordonate de vertexuri, culori, normale la suprafețe şi flag-uri de muchie). Fiecare vertex şi datele asociate se procesează independent, secvențial şi în acelaşi mod, singura excepție de la regulă fiind întrebarea dacă grupul de vertexuri trebuie decupat astfel încât o primitivă anume să se încadreze într-o regiune specificată; în acest caz se pot modifica datele vertexului sau se pot crea noi vertexuri, tipul de decupare depinzând de primitiva pe care o reprezintă grupul de vertexuri.

Comenzile sunt întotdeauna procesate în ordinea primirii, deşi poate exista o întârziere nedeterminată înainte de intrarea în joc a unei comenzi; acest lucru înseamnă că fiecare primitivă este complet desenată înainte ca o comandă următoare să devină efectivă şi ca toate comenzile pentru interogarea stării să returneze date care sunt compatibile cu execuția completă a tuturor comenzilor OpenGL precedente.

OpenGL oferă programatorului control total asupra operațiilor fundamentale, care se pot executa pe obiecte grafice bi- sau tridimensionale, aceasta incluzând specificații asupra unor parametri cum ar fi transformări de matrice, operatori de actualizare a pixelilor etc. Trebuie spus că nu se oferă totuşi metode pentru prelucrarea formelor de complexitate ridicată, astfel că uneltele OpenGL mai curând specifică cum se ajunge la un anumit rezultat, decât arată înfățişarea finală a rezultatului; de aceea spunem că OpenGL este mai mult procedural decât descriptiv, lucru care incumbă cunoaşterea bună a metodologiei.

Modul în care este interpretat OpenGL este de tip client-server, adică o aplicație: când emite o comandă, primeşte răspuns de la mecanismul OpenGL, adică de la server. Interesant este că acest echipament poate sau nu să ruleze pe unul şi acelaşi calculator cu clientul: prin urmare putem spune că OpenGL este transparent în mod rețea. Un server poate menține mai multe contexte OpenGL, fiecare din ele fiind o stare grafică încapsulată, iar un client se poate conecta oricând la oricare din ele, printr-un protocol Windows îmbogățit sau printr-unul nou creat. Nu există comenzi OpenGL care să permită preluarea unei intrări de la client.

Efectele comenzilor OpenGL asupra spațiului cadru sunt, finalmente, controlate de către sistemul de ferestre, care se ocupă cu alocarea de resurse destinate acestuia.

Cu alte cuvinte, sistemul de ferestre îşi dă seama care zone ale spațiului cadru pot fi accesate de către OpenGL la un moment dat şi comunică acest lucru serverului OpenGL.

Operații OpenGL de bază

Comenzile OpenGL intră pe rând într-o conductă de comenzi, unele comenzi specificând obiectele geometrice, care se vor crea sau prelucra, altele specificând operațiile concrete la care se va face apel în cadrul diverselor niveluri de procesare. Trebuie spus că nu este obligatorie trimiterea automată a tuturor comenzilor prin conducta de comenzi, ci se poate alege stocarea unui bloc de comenzi într-o listă de desfăşurare, care va fi analizată ulterior. Avem câteva etape de trecut în revistă în cele ce urmează.

Evaluarea - este etapa care oferă mijloace eficiente pentru aproximarea curbelor şi a geometriei suprafețelor, prin evaluarea comenzilor polinomiale ale datelor de intrare.

Asamblarea primitivelor şi operații pe vertexuri – este etapa în cadrul căreia OpenGL procesează primitivele geometrice, adică puncte, segmente şi poligoane, care toate sunt definite prin vertexuri. Acestea din urmă sunt transformate şi iluminate, iar primitivele sunt decupate şi preparate pentru următoarea prelucrare.

Rasterizarea – este operația prin care se produc (se generează) o serie de adrese şi valori asociate în spațiul cadru, folosind o descriere bidimensională a primitivelor. Fiecare fragment astfel produs este introdus în faza următoare.

Operații pe fragment – sunt cele care preced transformarea finală în pixeli a datelor utilizate. Amintim aici actualizarea spațiului cadru prin luarea în calcul a valorilor pentru axa Z, stocate anterior şi amestecarea per-pixel a culorilor.

Datele de intrare pot fi mai curând pixeli decât vertexuri, caz în care se poate trece direct la operația de prelucrare direct pe pixel, rezultatul fiind stocat ca textură în memorie, pentru a se folosi în procesul de rasterizare sau, la fel de bine, fiind rasterizat, după care fragmentele rezultate sunt introduse în spațiul cadru ca şi cum ar fi fost generate din date geometrice.

Programatorul are acces la toate componentele OpenGL, inclusiv la conținutul locațiilor de memorie unde se află textura (specificăm faptul că, prin termenul de textură, vom înțelege modelul tip „covor" care defineşte o imagine) şi chiar la spațiul însuşi.

Vedere de ansamblu pe operații şi comenzi

Multe comenzi se ocupă de desenarea concretă a primitivelor, altele de controlul asupra modalității şi condițiilor în care se desfăşoară desenarea, altele de interacțiunea cu spațiul. În cele ce urmează, ne vom concentra asupra a 5 categorii de operațiuni, aşa cum vom vedea mai jos.

Conducta de procesare - aici avem 3 tipuri de date de intrare: vertexurile şi 2 tipuri de date asociate, adică setul de culori şi forma texturii. Vom reține că vertexurile sunt descompuse în primitive, apoi acestea sunt transformate în fragmente, apoi acestea în pixeli. Nu trebuie uitat că, în cele ce urmează, vom utiliza doar prima parte a numelui unei funcții OpenGL, pentru a creşte gradul de simplificare. De exemplu, funcția glVertex2f() primeşte ca argumente 2 numere reale pe 32 biți fiecare (x,y), în timp ce glVertex3fsv() primeşte 3 argumente de tip întreg pe 16 biți fiecare (x,y,z), astfel că folosirea formulării glVertex*() le are în vedere pe ambele variante.

4. Structura datelor

Informațiile utilizate în realizarea ***reconstituirii 3D*** sunt structurate pentru crearea bazei de date *orientată pe obiecte*. Modelarea propriu-zisă constă în definirea claselor de obiecte, ale atributelor lor şi a relațiilor între obiecte. Organizarea cunoştințelor referitoare la ***reconstituirea 3D*** este făcută conform modelului descris mai jos [20],[21],[18].

Baza de date a aplicației stochează structura completă a elementelor arhitecturale componente, care să permită corespondența dintre diversele primitive şi piesele arhitecturale, precum şi cea dintre diverse primitive. Primitivele sunt de 3 tipuri: punct, segment, poligon, de tip triunghi şi patrulater.

Baza de date este constituită din tabele referitoare la elemente decorative, primare şi de legătură.

În continuare, este prezentată structura acestor tabele.

- Tabelul cu **piesele arhitecturale componente**; posedă câmpurile de mai jos.
  - număr identificare,
  - nume,
  - amplasare pe axa OX,
  - amplasare pe axa OY,
  - amplasare pe axa OZ,
  - lungime pe axa X,
  - lungime pe axa Y,
  - lungime pe axa Z.
- Tabel cu **elementele constitutive** primare de tip punct; posedă câmpurile de mai jos:
  - număr identificare,
  - amplasare pe axa OX,
  - amplasare pe axa OY,
  - amplasare pe axa OZ,
  - valoarea componentei cromatice de roşu,
  - valoarea componentei cromatice de verde,
  - valoarea componentei cromatice de albastru,
  - valoarea coeficientului de strălucire.
- Tabel cu **elementele constitutive primare de tip segment**; posedă câmpurile de mai jos;

- număr identificare,
- amplasare pe axa OX a capătului unu,
- amplasare pe axa OY a capătului unu,
- amplasare pe axa OZ a capătului unu,
- amplasare pe axa OX a capătului doi,
- amplasare pe axa OY a capătului doi,
- amplasare pe axa OZ a capătului doi,
- valoarea componentei cromatice de roşu,
- valoarea componentei cromatice de verde,
- valoarea componentei cromatice de albastru,
- valoarea coeficientului de strălucire,
- valoarea grosimii segmentului.

Tabel cu elementele constitutive primare de tip triunghi; posedă câmpuri care memorează informaţie referitoare la: amplasarea în spaţiu (cele 9 coordonate), cromatica interiorului (R,G,B, ALFA).

Tabel cu elementele constitutive primare de tip patrulater; posedă câmpuri care memorează informaţie referitoare la: amplasarea în spaţiu (cele 12 coordonate), cromatica interiorului (R,G,B, ALFA).

În plus, există **tabele de legătură**, după cum urmează:

- între piese arhitecturale şi segmente,
- între piese arhitecturale şi poligoane de tip triunghi,
- între piese arhitecturale şi poligoane de tip patrulater,
- triunghi-patrulater.

Realizarea reconstituirii 3D constă în:

- descompunere a fotografiei clădirii;
- reconstituire 3D a imaginii.

Pentru funcţionarea software-ului de reconstituire 3D, pe CD va fi instalat plug-in-ul Blaxxun Contact, care asistă navigatorul de Internet în interpretarea codului vrml, adică modul cum să preia vertexurile (colecţiile de puncte 3D) către funcţiile OpenGL ale sistemului Windows. În procesul de reconstituire 3D a clădirilor dispărute, se foloseşte scriptul vrml (Virtual Reality Modeling Language, limbaj pentru modelarea realităţii virtuale). În principiu, un fişier care include cod vrml are extensia "wrl" şi reprezintă o suită de obiecte elementare, aflate la anumite coordonate, având anumite dimensiuni; fiecare faţă plană este colorată într-o anume culoare sau este acoperită de către o imagine (numită textură) şi este încărcată dintr-un fişier de tip bmp, gif, jpg.

Descompunere a fotografiei clădirii

În procesul de reconstituire, se porneşte de la o fotografie de epocă, reprezentând Casa Moruzzi (descrierea acesteia este în figura 1), referinţă arhitectonică pentru urbanistica Bucureştilor din a doua jumătate a secolului al XVIII-lea. Pentru început, au fost extrase imaginile elementelor arhitecturale componente (pereţi, acoperiş, nişe etc.) şi au fost prelucrate astfel încât să se obţină imaginea din vedere frontală. Acest lucru a fost necesar pentru a obţine o redare exactă a fiecărei feţe, indiferent de poziţia unghiulară a ansamblului.

Ilustrăm procesările acestei etape prin exemplele de mai jos.

Figura 1 Casa Moruzzi - demolată în anii 1930. Fotografia datează din anul 1920

În figura de mai jos, sunt prezentate imaginile elementelor arhitecturale componente (pereți, acoperiș, nișe etc.), extrase din fotografia de studiu, și „frontale".

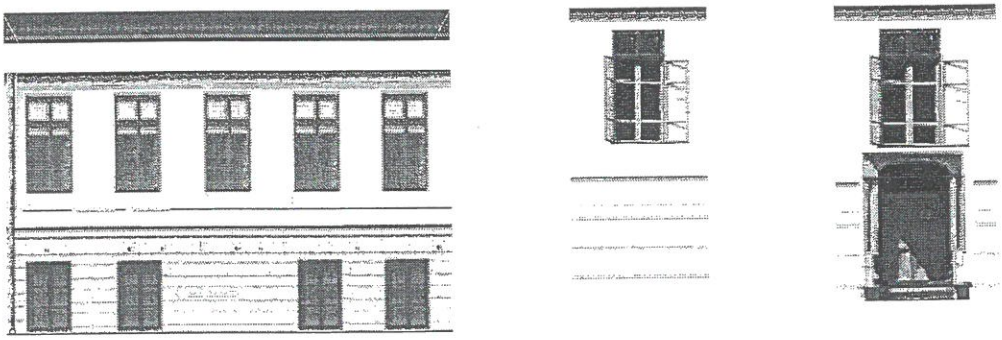


Figura 2. Streașina frontală, peretele frontal, peretele semihexagon lateral și respectiv peretele semihexagon lateral invizibil – prelucrate astfel încât să apară exact în proiecție ortogonală

În ceea ce privește părțile care nu se văd în fotografia originală, s-a plecat de la supoziția unor simetrii geometrice, astfel că ansamblul a fost reconstituit pe baza elementelor arhitecturale vizibile în imagine.

Am optat, în continuare, pentru amplasarea casei pe o peluză, pe care am conceput-o ca în figura de mai jos.

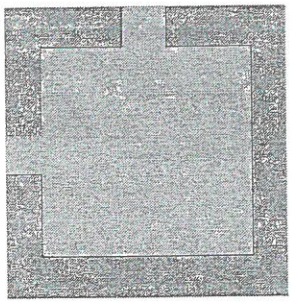


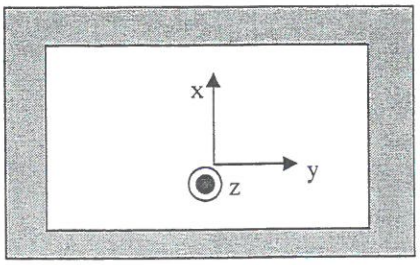
Figura 3. Peluza

Pentru a evita golurile de imagine, în timpul rotirii figurii, s-a aplicat o textură omogenă pe normala inversă a planului ce reprezintă peluza.

Este esențial de menționat că, între toate aceste imagini, s-au impus raporturi dimensionale 2D adecvate, astfel încât clădirea reconstituită să fie cât mai apropiată din punct de vedere al proporționalității de clădirea originală.

Reconstituirea 3D a imaginii

În continuare, s-a procedat la stabilirea coordonatelor fiecărei părți componente a clădirii, plecând de la ideea că centrul ecranului coincide cu centrul sistemului cartezian de coordonate 3D, ca în desenul de mai jos:



Trebuie spus că, scriptul vrml acceptă rotirea, translatarea şi scalarea sistemului de coordonate. Pentru iniţializarea dimensiunilor, am plecat de la înălţimea unui om (prezent în figură), pe care am aproximat-o la 1.75.

Pentru a accelera funcţionarea, vom împărţi toate coordonatele la coordonata cu valoarea absolută maximă.

Redăm mai jos câteva exemple de cod vrml.

```
#VRML V2.0 utf8

Group
{
   children
   [
      Transform #iarba
      {
        children Shape
        {
            appearance Appearance
            {
               texture ImageTexture { url "iarba.jpg" }
               material Material{ diffuseColor 0 0 1 }
            }
            geometry IndexedFaceSet
            {
               coord Coordinate
               {
                  point
                  [
                  -1.038168, 0.000000, 0.908397,
                    2.450382, 0.000000, 0.908397,
                    2.450382, 0.000000,-2.908397,
                  -1.038168, 0.000000,-2.908397
                  ]
               }
               coordIndex [ 0 1 2 3 ]
               texCoord TextureCoordinate
               {
                  point [ 0 0, 1. 0, 1. 1., 0 1. ]
               }
            }
        }
      }
      Transform #perete 1, deci frontal
      {
        children Shape
        {
            appearance Appearance
            {
               texture ImageTexture
               {url "zid_frontal_fara_streasina_fata.jpg" }
                material Material{ diffuseColor 1 0 1 }
```

```
            }
            geometry IndexedFaceSet
            {
               coord Coordinate
               {
                  point
                  [
                  00.000000, 00.000000, 00.000000,
                  01.412214, 00.000000, 00.000000,
                  01.412214, 00.820611, 00.000000,
                  00.000000, 00.820611, 00.000000
                  ]
               }
               coordIndex [ 0 1 2 3 ]texCoord TextureCoordinate
               {
                point [ 0 0, 1. 0, 1. 1., 0 1. ]
                }
            }
         }
      }
   ...
      Background { skyColor 1. 1. 0.9 }
      NavigationInfo { type "EXAMINE" }
   ]
}
```

Observăm aici câteva apariții de funcții:

- **Transform**, definește o colecție de colecții de figuri plane;
- **children Shape**, definește o colecție de figuri plane, care sunt umplute cu o culoare implicită și /sau sunt acoperite cu o textură (adică, cu imaginea dintr-un fișier bmp, gif, jpg etc);
- **appearance Appearance**, stabilește textura și / sau culoarea de umplere;
- **material Material** {diffuseColor 0 0 1}, stabilește culoarea de umplere: aici, de exemplu, combinația cromatică este Roșu=0*255, Verde=0*255, Albastru=1*255;
- **geometry IndexedFaceSet**, găzduiește colecția de puncte în coordonate 3D și sensul în care eventuala textură este parcursă pentru a fi asociată corect cu figura plană;
- **point**, memorează colecția de puncte în coordonate 3D;
- **coordIndex**, memorează sensul de parcurgere a texturii;
- **texCoord TextureCoordinate**, memorează scalarea texturii;
- **NavigationInfo**, stabilește modalitatea în care imaginea este mișcată în ecran la acționarea mouse-ului sau tastelor.

Trebuie menționat că, pentru a nu avea goluri de imagine, figurile expuse unor astfel de situații trebuie să aibă textura sau culoarea pe ambele normale ale suprafeței.

Scriptul vrml oferă posibilitatea reutilizării codului, prin folosirea claselor derivate. O clasă derivată suportă adăugiri de funcționalitate.

O altă facilitate pusă la dispoziția utilizatorului este lucrul cu obiecte de tip cronometru, care stabilesc momentul și viteza de desfășurare a unor evenimente (care, la rândul lor, pot fi declanșate de la terminale) cum ar fi: deschidere / închidere de uși, ferestre etc.

În încheierea materialului, prezentăm 2 capturi de ecran extrase din funcționarea concretă a fișierului de tip vrlm.

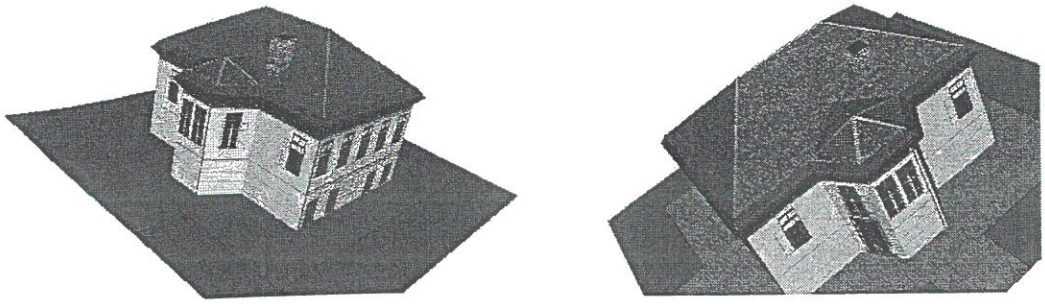


Figura 4. Casă reconstituită

5. Arhitectura hardware şi software

Tehnologia de realizare a **reconstituirii 3D** presupune o arhitectură *de tip multimedia*, care să gestioneze imagini 3D, sunet.

În cadrul acestei lucrări, s-au analizat diferite produse software, care realizează generarea unei imagini 3D pornind de la fotografii vechi.

- ANSYS oferă o soluţie pentru crearea meşurilor conţinând instrumente puternice pentru preprocesarea şi postprocesarea meşurilor generate ale oricărei figuri geometrice, pentru a produce aproape orice tip de elemente, pentru orice formă fizică, pentru realitatea virtuală sau aplicaţii;
- CUBIT, oferit gratuit pentru cercetare, se bazează pe algoritmi de generare a meşurilor hexaedrale şi conţine module de mapare şi netezire a meşurilor.

În urma analizei algoritmilor de generare a meşurilor, produsul software cel mai adecvat proiectului de reconstituire 3D este CUBIT.

Realizarea CD multimedia de **reconstituire 3D** presupune manipularea unui volum mare de informaţii pentru stocarea imaginilor procesate 3D - ceea ce impune utilizarea următoarei configuraţii hardware şi software:

- calculator PC Pentium IV, DDRAM 512MB, HDD 60GB, placă video GeForce 128MB, CD-Rom, CD Writer, placa de sunet Creative Audigy, microfon şi căşti profesionale (de studio);
- Sistem Operare **Win 2000** sau **Win XP**;
- **Adobe Photoshop 7**;
- software **OpenGL** ce permite virtualizarea imaginii;
- MySQL server pentru gestionarea bazei de date.

6. Concluzii

În cadrul lucrării, a fost realizat un instrument de **Reconstituire 3D** cu scopul de a reconstitui tridimensional clădiri din fotografii vechi. Componenta de **Reconstituire 3D** va completa CD-ul multimedia şi va permite vizualizarea în Internet 3D unor clădiri create după descrieri de secolul al XIX-lea şi începutul secolului XX.

În prezent, ***sistemele multimedia*** alături de reţeaua Internet, au revoluţionat circulaţia informaţiei. După realizarea paginii proprii în reţeaua Internet pentru ***Muzeele din România*** şi ***Compact Disc-ului Interactiv*** de Reconstituire 2D, este importantă continuarea susţinerii accesului la patrimoniului cultural prin crearea unui spaţiu virtual 3D.

Realizarea acestei teme a fost posibilă în urma acordului dintre Institutul Naţional de Cercetare-Dezvoltare în Informatică, ICI, Bucureşti şi Muzeul Municipiului Bucureşti.

Continuarea lucrării

Pe viitor, se vor dezvolta algoritmi pentru realizarea reconstituirii 3D şi animaţiei vizualizării interiorului unei clădiri din Bucureştii sfârşitului de secol XIX.

Bibliografie

1. **OWEN, S. J.**: A Survey of Unstructured Mesh Generation Technology. În: Proc. of the 7th International Meshing Roundtable, pp. 239-267, http://www.andrew.cmu.edu/user/sowen/survey, 1998.
2. **OWEN, S. J.**: Meshing Software Survey, Structured Grid Generation Software, web page: http://www.andrew.cmu.edu/user/sowen/software/structured.html
3. **GEORGE, P.L., HECHT, F., E. SALTEL**: Automatic Mesh Generator with Specified Boundary", Computer Methods in Applied Mechanics and Engineering, North-Holland, 1991, vol 92, pp. 269-288.
4. **LOHNER, R.**: Progress in Grid Generation via the Advancing Front Technique. În: Engineering with Computers, 1996, vol 12, pp. 186-210.
5. * * *: **TetMesh**, GSH3D web site: http://www.simulog.fr/tetmesh/
6. * * *: **ANSYS** web site: http://www.ansys.com
7. * * *: **CUBIT** Mesh Generation Toolkit, web site: http://cubit.sandia.gov/
8. **WHITE, D. R.**: Automated Hexahedral Mesh Generation by Virtual Decomposition. În: Proc. ff the 4th International Meshing Roundtable, Sandia National Laboratories, 1995, pp. 165-176.
9. **STATEN, M. L., S. A. CANANN, OWEN, S.J.**: BMSWEEP: Locating Interior Nodes During Sweeping. În: The 7th International Meshing Roundtable, 1998.
10. **OWEN, S. O, STATEN, M. L., Scott A. CANANN, S. A., Sunil SAIGAL** :Advancing Front Quad Meshing Using Local Triangle Transformations. În: Proc. Of the 7th International Meshing Roundtabl, 1998.
11. **NOWOTTNY, D.**: Quadrilateral Mesh Generation via Geometrically Optimized Domain Decomposition. În: Proc. Of the 6th International Meshing Roundtable, 1997, pp. 309-320.
12. **WHITE, D. R., KINNEY, P.**: Redesign of the Paving Algorithm: Robustness Enhancements through Element by Element Meshing. În: Proc. Of the 6th International Meshing Roundtable, Sandia National Laboratories, 1997, pp. 323-335.
13. * * *: **MacNeal-Schwendler Home Page**, web site: http://www.mscsoftware.com/
14. * * *: **FEGS web site**: http://fegs.co.uk
15. * * *: **Virtualizarea Realității și Vizualizarea: Home page**: http://iit-iti.nrc-cnrc.gc.ca/r-d/3d-vir-reality-realite-vir-3d_e.html
16. * * *: **3D From paintings and photos** : http://iit-iti.nrc-cnrc.gc.ca/projects-projets/paintings-tableaux_e.html
17. **EL-HAKIM, S.F., J-A., BERALDIN, M. PICARD:** 3D Modeling of Heritage Monuments. GIM International, 17(4): 13-15. April 2003. NRC 45821.
18. **EL-HAKIM, S.F.**: Semi-automatic 3D Reconstruction of Occluded and Unmarked Surfaces from Widely Separated Views," Proceedings of ISPRS Commission V Symposium, Close Range Visualization Techniques, Corfu, Greece. September 1-2, 2002. pp. 143-148 NRC 44944.
19. * * *: Delphi - A Guide to Programming
20. * * *: Adobe Photoshop
21. * * *: SQL server
22. **CIOCOIU, L., A-M. BOROZAN, C. COȘOIU**: Tema A19 / 2001 – "Muzeu virtual privind arhitectura până la începutul sec. XX în rețeaua Internet".
23. **CIOCOIU, L., A-M. BOROZAN, C. COȘOIU**: Tema A20/2001 – "Arhive virtuale specifice muzeelor județene în rețeaua Internet".