Universidad del País Vasco Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

INDUSTRIA TEKNOLOGIAREN
INGENIARITZAKO GRADUA

GRADU AMAIERAKO LANA

PROZEDURA BATEN MATLAB BIDEZKO GARAPENA INGERADEN INTERPRETAZIOAREN OPTIMIZAZIO TOPOLOGIKOAN OINARRITUTA

Ikaslea: Murua De la Mata, Oihane
Zuzendaria: Ansola Loyola, Ruben

Ikasturtea: 2018-2019

Data: Bilbo, 2019ko Uztailaren 22a

Laburpena:

Proiektu hau egituren diseinuan erabilitako optimizazio topologikoaren ingeraden azterketan oinarritzen da. Jarraituko den prozedura aurrez ezarritako karga eta murrizketa batzuen pean, hasierako egoeratik abiatuz nahi den bolumen murrizketa lortzean geratzen den egituraren topologia bilatzean datza, beti ere osagaien propietate mekanikoak mantenduz. Lana zati desberdinetan banatu daiteke, hasteko optimizazio topologikoaren analisi teorikoa eta emaitzen azterketa egingo da, Matlab kode baten bitartez. Emaitza hauen analisia egitean proiektuaren helburua dentsitate aldaketako guneen arteko, hau da, zonalde zuri (dentsitatea 0 dena) eta beltzak (dentsitatea 1 dena) banatzen dituen, ingeradak lortzea da. Horretarako Elementu Finituen Metodoa (EFM) oinarritzat hartuko da, proiektuaren aldagai nagusia lortutako eremua banatzen duen elementu finitu bakoitzaren dentsitatea daukan matrizea izango da eta. Bukatzeko, dentsitate matrize honetatik lortutako ingeradekin CAD artxibo bat sortu eta 3 dimentsiotan inprimatuko da proiektuaren egitura.

Resumen:

Este proyecto consiste en el estudio de los contornos de la optimización topológica para el diseño de estructuras. El procedimiento a seguir consiste en encontrar la topología que mantenga las propiedades mecánicas de la estructura ante unas cargas y restricciones preestablecidas, a partir de un dominio inicial dado, disminuyendo el volumen de materia hasta lo deseado. El trabajo se puede dividir en varias partes, empezando por el análisis teórico e interpretación de resultados de la optimización topológica mediante un código de Matlab. Al hacer el análisis de estos resultados, el objetivo del proyecto es conseguir los contornos que sucede el cambio de densidades, esto es, los contornos que dividen las zonas blancas (de densidad 0) de las negras (de densidad 1). Para esto, se emplea el Método de Elementos Finitos (MEF), ya que la variable principal del proyecto es la matriz que contiene el valor las densidades de cada elemento finito en que se divide el dominio conseguido. Para terminar, se creara un archivo CAD con los contornos conseguidos de la matriz de densidades y la estructura se imprimirá en 3 dimensiones.

Abstract:

This Project is based on the topological optimisation study of the contours used for structural design. The procedure involves the search of a topology that maintains the mechanical properties of the structure under the effects of loads and pre-established constraints, starting from an initial domain and diminishing the volume of the material used. This Project is divided into several parts. It begins with the theoretical analysis and interpretation of the topological optimisation results via Matlab code. The objective of this analysis is to obtain the contours resulting from the density difference; that is to say, the contours that separate the white areas (density value 0) from the

black zones (density value 1). The domain is divided into finite elements, the density value of which is arranged into a matrix. Consequently, the Finite Element Method (FEM) is used. At the end of the process, a CAD file containing the contours obtained from the density matrix will be created and 3D printing methods will be used so as to generate the physical representation of the structure.

Hitz gakoak: *Optimizazio topologikoa, ingerada, Matlab kodea, bolumen murrizketa, diseinu aldagaiak, CAD artxiboa, 3D inpresioa.*

Palabras clave: *Optimización topológica, contorno, código de Matlab, restricción de volumen, variables de diseño, archivo CAD, impresión 3D.*

Key words: *Topological optimization, contour, Matlab code, volume constraint, design variables, CAD file, 3D printing.*

AURKIBIDEA

IRUDIEN AURKIBIDEA

TAULEN AURKIBIDEA

1 SARRERA

Proiektua hau gaur egun modan dagoen eta geroz eta gehiago erabiltzen den optimizazio topologiko bidez garatutako egitura edo piezak lortzean oinarritzen da. Hauek normalean forma arraro edo konplexuak dituztenez zenbait pausu jarraituz lortzeko bidea egingo da fabrikazio prozedura berritzailea, 3D inpresioa, martxan jarriz.

Horretarako egungo ingeniaritza munduan oso garatuta dagoen software bat erabiliko da, Matlab izenekoa eta hainbat eta hainbat gai landu baitaitezkeena. Izan ere, ingeniaritza mundua oso zabala da eta arlo asko erlazionatzen dira. Beraz, oso software erabilgarria da aipatutako ezaugarri hauek biltzen dituelako.

Lan honetan Matlab softwarearen kode bat erabiliz hasten da, kode hau *'A 99 line topology optimization code written in Matlab'* izenez ezagutzen da. Prozedura ezberdinak jarraituz, lanean zehar azaltzen direnak, CAD artxibo bat lortzera iritsiko da. Azken motako artxibo honekin, 3D inpresioa gauzatu daiteke inprimagailua honekin bateragarria delako.

Azken urteetan oso zabaldua dago 3D inpresioa erabiliz egitura edo pieza ezberdinak lortzea, honela, material kantitatea aurreztea gertatzen da besteak beste. Materialaren aurrezte hau optimizazioaren eskutik doa. Optimizazioak egitura onena lortzea helburu duen aldetik, ingeniarientzat arlo erakargarria da mugarri berriak jartzen dituelako. Modu honetan, mundua garatzen doan heinean, ingeniaritza garatzen doa eta geroz eta gauza hobeak lortzeko gai da. Gainera fabrikazio gehigarriari esker optimoak diren egiturak sortzeko aukera errazten da. Honek material metalikoak erabiltzea ahalbidetzen du inprimatzerakoan, gaur egun arte pentsaezina zena.

Atal honekin amaitzeko, aipatu behar da jarraituko den egitura lana jorratuko den testuingurua azalduz hasiko dela, eta helburuak eta isipenarekin jarraitu. Segidan lanaren onura batzuk aipatuko dira, eta ondoren lanaren atal garrantzitsuenetako bat, optimizazio topologikoaren problema eta horretan oinarritutako Matlab kodearen inplementazioa. Jarraian optimizazio topologiko honen interpretazioa egingo da, baita Matlab eta excel bidezko lotura azaldu interpretazioa osatzeko. Proiektuarekin amaitzen joateko CAD softwarearen inplementazioa azalduko da, pieza inprimatzeko

bidea argiago ikusiz. Guzti hau biltzen duen Gantt diagrama eta planifikazioaren azalpena egingo da hurrengo atalean. Bukaera ematen joateko, lanean zehar lortutako emaitza eta elkarren arteko konparaketak azaltzen dira. Lana borobiltzeko landutako ororen ondorioak ateratzen dira, eta etorkizunerako ekarpenak ere baloratzen dira. Era berean, azkeneko atalak bibliografia eta lana osatzeko beharrezko eranskinak dira.

2 TESTUINGURUA

Lan hau Matlab bidezko prozedura baten garapenetik hasiko da eta orokorrean optimizazio topologikoaren ingeraden interpretazioan oinarrituko da, atal honetan zehazki testuingurua azalduko da.

Optimizazio topologikoak material kantitate bat azalera edo bolumen zehatz bat betetzeko banaketa optimoa bilatzen duen prozesua deskribatzen du. Prozesu honek egitura arinena bilatzea du helburu nagusia, beti ere propietate eta betebehar mekanikoak mantenduz.

Garai batean egituren diseinua gauzatzeko arma eskuzko kalkulu bidezkoa zen, honek ordea urteen poderioz, eta batez ere ordenagailuen agerpenari esker, erabateko bilakaera izan du. Ordenagailuekin FEM (‘*Finite Element Method*’) erabiliz optimizazioa askoz ere modu errazago eta azkarragoan egitea lortzen da. FEM ingeniaritza problemak askatzeko zenbakizko metodo konputazional bat da. Horregatik, optimizazioa bezalako diseinu eta kalkulu konplexuak egiteko oinarrizko erreminta kontsideratzen da.

2.1 OPTIMIZAZIOA

Optimizazio kontzeptuak arazo baten aurrean soluzio posible egokiena aurkitzean datza. Soluzio posible egokiena esaten da hoberena ez baita bilatzen dena, optimoena baizik. Azken batean, emaitza egokiena beharrezko funtzioa betetzeko ezaugarri hoberenak dituena izango da.

Ondo desberdindu beharrekoak dira optimizazio eta efizientzia gaiak. Izan ere, maiztasun handiarekin nahasten dira. Optimizazioari begira etengabeko hobekuntza erakusten du eta aurkezten duen erantzuna onena dela esan daitekeela uste da. Baina hau ezta horrela, efizientziari begira ikusten baita egiturarik optimoena ez dela beti errentagarria. Edozein hobekuntzak kostu bat baitauka, eta kostu hau errentagarria dela esango da hobekuntza horren kostua lortutakoaren kostua baino txikiagoa baldin bada [1].

Aipatzekoa da, diseinu berrien garapenean gehien erabiltzen den erremintetako bat dela egituren optimizazioa, etengabeko garapenean dagoen arlo bat delarik. Gainera,

optimizazioak topologiaren aldetik begiratuz fabrikazio gehigarriarekin duen lotura nabarmenaz, edozer eraikitzeko aukera ematen du.

2.2 EGITUREN OPTIMIZAZIO MOTAK

Egituren optimizazioa hiru sail edo mota nagusitan banatu daiteke, tamaina, forma eta topologiari dagokienak hain zuzen ere.

2.2.1 Tamainaren optimizazioa

Mota hau egituraren elementuen neurri optimoak zehaztean datza beti ere duten forma mantenduz, adibide gisa barren sekzioak edo plaken lodiera aldaketak aipatu daitezke. Hemen kokatuko lirateke optimizazio problema errazenak, izan ere, aldagai kopurua mugatua da.

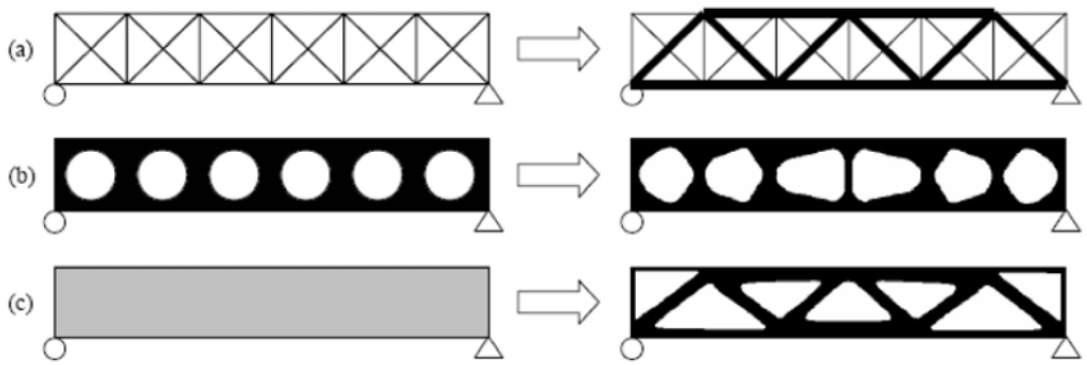
2.2.2 Formaren optimizazioa

Formaren optimizazioari dagokionez, honen helburu nagusia izenak dioen moduan egitura topologiko zehatz baten ingeraden forma optimoa lortzea da. Kasu honetan, aurredefinitutako forma batetik abiatzen da eta optimizatzeari ekiten zaio egituraren kanpoko eta barneko ingeradak aldatuz.

2.2.3 Optimizazio topologikoa

Egitura bateko materia banaketa optimoa bilatzean datza azken mota honek. Hasierako materia domeinu batetik abiatuta, euskarri finko eta jasan beharreko kargen espezifikazioen funtzioan, materiaren banaketa hoberena ezartzen da karga jakin batzuen pean deformazio eta esfortzuak minimora jaitsiz.

Azpiko irudian (Irudia 1) egituren hiru optimizazio motak ikusten dira. Azken optimizazio mota honek forma eta tamainako optimizazio klasikoekin alde handia dauka, azken soluzioak hasierako diseinuarengan eragin nabarmena duelako. Optimizazio topologikoak hasierako datu minimo batzuk behar ditu, honek optimizazioa zeharo indartsua bihurtzen du egituren sorkuntzari dagokionez [2].



Irudia 1. Optimizazio motak: (a) Tamaina, (b) Forma eta (c) Topologia.

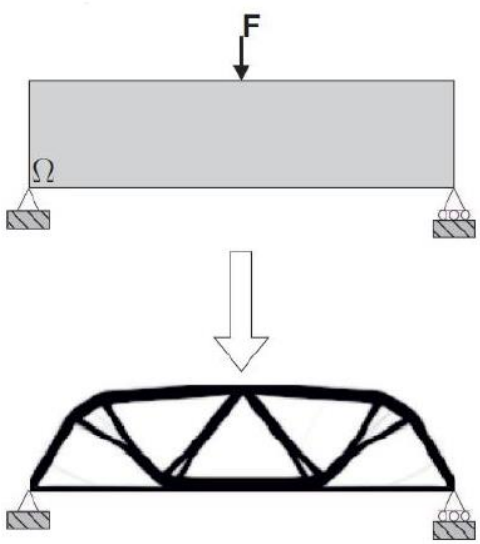
2.3 OPTIMIZAZIO TOPOLOGIKOA: DEFINIZIOA ETA AURREKARIAK

Modu orokorrean azalduz, optimizazio topologikoa ordenagailu bidezko planteamendu bat da. Planteamendu honek materialaren banaketa optimizatzen du diseinuko domeinu eta ingurune baldintzak kontuan izanik [3].

Optimizazio topologikoaren inbestigazioa etengabe eta oso azkar garatzen ari den mekanika konputazionalaren alorra da. Honek matematikari eta ingeniari mekanikoengan interesa piztu du, azken urteetan oso zabaldua eta landua izatera iritsiz. Optimizazio topologiak aplikazio interesgarriak dauzka mekanika, multifisika eta micro- eta nanoteknologietan, diseinu eraginkorrak aurretiazko gutxieneko suposaketak erabiliz lortu daitezkeelako. Gainera, fabrikatzen diren piezek ahalik eta pisu gutxien eduki eta erresistentzia nahikoa eutsi behar duten kasuetan, mekanika aeroespazialean esaterako, erabiltzen da optimizazio topologikoa industrian. Azken kasu honetan diseinatzaileak erabakitzen du murriztutako pisua, horretarako erresistentzia entseguak egiten direlarik baldintza guztiak betetzen dituela ziurtatzeko [4].

Aurretik aipatu bezala optimizazioan emandako lehen pausuak eskuzko diseinu bidezkoak izan ziren, baina ordenagailuen (FEM) agerpenaren ondorengo ibilbidea honakoa izan da. Optimizazio topologikoaren arlo honetan egindako lehenengotariko aplikazioa karga puntualen eraginpean zurruntasun maximoa eta mugapen zehatzak (euskarriak) betetzen zituen egituren diseinua izan zen. Azpiko irudiak (Irudia 2) erakusten du F izeneko karga puntual bat eusten duen domeinu bidimentsionala (Ω), hau beheko iskinetan euskarri artikulatuen gainean kokatzen delarik. Irudiaren azpiko

aldean optimizazio problemaren erantzuna dago, honek aurkeztutako baldintzen eta bolumen murrizketaren aurrean lortzen dena.



Irudia 2. Optimizazio topologikoa karga baten pean.

Optimizazio topologikoaren metodoak diseinu estrukturalaren problemari zein diseinuko domeinuaren material banaketa optimoari heltzen dio, sistemaren errendimendu hoberena eskaintzen duen geometria zehaztea helburu duelarik. Honek modu automatikoan arruntak ez diren egitura eta elementu mekanikoak sortzeko aukera ematen du. Ingeniariaren trebetasun eta esperientzia elementu konplexuak eratzeko oinarrizkoak izan dira betidanik. Hala ere, azken urteetan egiaztatu da topologiaren optimizazioan, ordenagailu bidezko diseinu optimoaren diziplinan, laguntza handikoa izan daitekeela egitura eta elementu mekanikoen diseinu optimoa. Optimizazio automatiko eta iteratiboaren bidez mota desberdinetako eginkizunen hobekuntzak asko errazten direlarik.

Optimizazio topologikoa gauzatzeko aukera desberdinak dauden arren, guztien helburua aurredefinitutako diseinu domeinuan materialaren distribuzio optimoa lortzea da.

Optimizazio topologikoaren inguruko lehen argitalpena, '*The limits of economy of material in framestructures'* izenburuduna, orain dela mende bat izan zen A.G.M Michell-en eskutik (1904) [4]. Michell ingeniariak barra giltzatuen egitura baten pisu murrizketa burutzeko optimizazio irizpide desberdinak aztertu zituen, emaitzak alde batetik hegaz zegoen habe bati aplikatuz. "Michell-en habea" izenez ezagutu zen hau,

alde batean landatua, besteko azpiko izkinan euskarri sinple batekin eutsia eta karga puntual bat landatutako aldeko goiko muturrean aplikatutako habe bezala sinplifikatu zen; honela optimizazio topologikoaren abiapuntu gisa erabili da.

Schmidt-ek proposatutako ideia iraultzaileak 60 hamarkadako optimizazio topologikoaren kontzeptuarengan eragin handia izan zuen, diseinu optimoaren lorpena ordenagailu bidezko prozedura iteratibo baten bidez lortzeko baseak ezarri zirelarik.

Hurrengo hamarkadan, 70.goan, optimizazio topologikoak garrantzi nabarmena irabazi zuen zientzia konputazionalen aurrerapenekin. Horregatik, Mröz eta Pragera autoreek optimizazio prozesuan zenbakizko metodoak erabiltzearen abantailez baliatu ziren.

Gaur egun ezagunena den optimizazio topologikoaren metodoa SIMP izenez ezagutzen dena da. SIMP siglek “Solid Isotropic Material with Penalization” adierazten dute eta Bendsøe zientzialariak 80. Hamarkadaren bukaeran (1989) proposatu zuen metodoa da. SIMP metodo honetako aldagaiak elementuaren dentsitate erlatiboak dira diskretizatutako diseinu domeinuan konstanteak direnak [5].

Jakina da soluzio optimoa optimizazio topologikoaren diskretizazio mailaren araberakoa dela, elementu finituen metodoan oinarritutako hainbat aplikaziotan ikusi den bezala.

Bendsøe berak eta Kikuchi-k urte bat lehenago (1988) mikroegitura eta homogenizazioan oinarritutako optimizazio topologikorako zenbakizko metodoak sakonki ikertu zituzten [4].

Azken urteotan sortutako beste metodo batzuk ere oso erabiliak izan dira, hauek optimizazio irizpidea (OC), asintota mugikorren metodoa (MMA) eta programazio lineal sekuentziala (PLS) dira besteak beste.

Optimizazio topologikoaren metodo hauek orokorrean diseinu problemak ebazteko erabili izan dira, beti ere fabrikazio prozesu tradizionalak, mekanizazioa edo fundizioa adibidez, kontuan hartuz. Teknika hauek fabrikazioan zenbait murrizketa dakartzate non sarritan kontuan izan behar diren diseinu eta optimizazio etapetan emaitza egokia lortu nahi bada.

Orain arte, aipatutako murrizketa hauek optimizazio topologikoaren metodoak aplikatzeko mugak ezarri behar zituzten, honela optimizazioaren diseinu eta fabrikazio erraztasunaren arteko konpromiso batera heltzera behartuz. Zentzu honetan bi aukera zituzten; alde batetik, murrizketak optimizazio problemaren formulazioan zuzenean sartzea, eta bestetik, problema gauzatu ostean murrizketak emaitzen interpretazio bidez lortzea. Lehen aukera nahiago izaten da gehienetan, baina murrizketak ezin dira optimizazio prozesuan errez sartu kasu guztietan [6].

2.4 OPTIMIZAZIO TOPOLOGIKOA ETA TARTEKO DENTSITATEEN INTERPRETAZIOA

Optimizazio topologiko bidez lortutako diseinuen fabrikazio errealaren beste aspektu bat domeinuaren tarteko dentsitateen interpretazioa da. Izan ere, oso ohikoa da metodo honen bidez lortzen diren soluzioetan. Diseinuko aldagaien tarteko balioak aurkezten dituzten ingurune hauen interpretazioa erabakigarria da ondorengo fabrikazioan, azken materialaren banaketan eragina izango duenez gero.

Urte askotan zehar sakontzen ibili den inbestigazio arloa aipatutako tarteko dentsitateak dituzten guneen agerpena azken soluzioan ekiditeko algoritmoen garapena izan da. Orain arte hauen fabrikazioa partzialki hurbiltzeko zaila izan dela medio. Horretarako normalean tarteko dentsitateen gabeziara bideratzen zuten penalizazio bidezko estrategietara jo da.

Hala eta guztiz ere, fabrikazio gehigarriaren teknologia berriak eskaintzen duten potentzialak aipatutako estrategiez baliatzeko eta komenientzia berriz planteatzeko beharrizana aurkeztu dute. Zentzu honetan, prozedura desberdinak proposatu dira non propietate eta tarteko dentsitateak dituzten tarteen portaera erreproduzitzea ahalbidetzen duten.

Etorkizun oparoa dakarren beste aukera bat kontzentrazio desberdinak dituzten materialen konbinazioa egin eta hauek inpresio bidez fabrikatzea da. Hauen bitartez ezaugarri mekaniko espezifiko batzuk dituzten konpositeak lortu daitezke.

Atal honekin amaitzeko, fabrikazio gehigarriak parametroen aldaketa prozesuan zehar kontrolatzeko aukera ematen duela esan behar da, beti ere puntu jakin batera arte. Esaterako eraikitako piezen porositatea kontrolatu dezake, hau materialaren propietate mekanikoekin erabat lotuta dagoela jakinda. Agerikoa da optimizazio

topologikoaren metodoetan planteamendu hauen integrazioak erronka eta aukera berriak suposatzen dituela. Aukera hauek material errekurtsoen aurreztearekin batera fabrikazio osagaien etekin handipena dakarte lortutako topologia zehaztasunez erreproduzitzeko [7].

2.5 OPTIMIZAZIO TOPOLOGIKOTIK 3D INPRIMAZIORA

Atal honetan optimizazio topologikoak 3D inpresioarekin daukan lotura azaltzen da, hau da, egitura konplexuak errealitate bihurtzeko bidea. Aipatzekoa da fabrikazio gehigarria guzti honekin erabat lotuta doala, izan ere diseinu optimo baten erreferentzia da forma arraroak eraikitzea ahalbidetzen duelako.

Fabrikazio gehigarria (Additive Manufacturing, AM) beste edozein fabrikazio prozesurekiko desberdina da kapaz kapa eraikitzen direlako piezak. Nahiz eta fabrikazio mota honen jatorria prototipoak egiteko modu azkarren merkatura esklusiboki zuzendutako teknologia izan (Rapid Prototyping, RP), gaur egun inbertsio eta ahalegin handia egiten ari da batez ere sektore aeronautikoan pieza erabat funtzionalak fabrikatzerakoan [8].

Egungo inbestigazioa material metalikoen fabrikazio gehigarriari zuzendutakoa izan da batez ere, gainera pieza metalikoak ekoizteko gai diren prozesu komertzialak existitzen dira. Ohikoena prozesu hauek bi taldeetan banatzea da: burdinurtu uzte bidezko moldeatzea (Fuse Deposition modelling, FDM) eta laser bidezko sintetizazio selektiboa (Selective Laser Meeting, SLM). Teknologia desberdinak diren arren, haien ezaugarrian diseinatzaileari diseinu askatasun esanguratsua ematen dio fabrikazio prozesu tradizionalekin alderatuz, honela egindako piezak optimizazio topologiko bidez lortutako diseinu optimora gehiago gerturatzen dira.

Guzti honek ingeniariari diseinua berriz ere planteatu eta osagai guztiei probetxua ateratzea eragiten du, hau izanik fabrikazio gehigarriak eskaintzen duen potentzial berria.

Fabrikazio tradizionalean baino diseinu muga gutxiago dituen arren, fabrikazio gehigarriaren praktikan zailtasunak daude jarraitu nahi diren optimizazio topologikoko prozedurak garatzean. Esan behar da, kasu askotan materialen gehikuntza teknologiak

erabiliz osagaiak egiteko egitura euskarri lagungarriak behar dira inpresioa gertatzen den bitartean [7].

Lan honetan hasiera batean, Matlab kodetik lortutako domeinu eta materia banaketa batetik abiatuz eta honi optimizazio topologikoa aplikatuz egitura optimoa lortzen da. Hala ere, hau ez da nahikoa hemen arazoa baitator, esan bezala egitura arraroa bada eraikitzeko zaila dela. Horregatik, aurrera pausu handia da optimoa den egitura horretatik CAD artxibo bat sortzea. Izan ere, 3D inprimagailua artxibo mota hau irakurtzeko gai da, eta beraz egitura optimoa lortzeko [3].

3 LANAREN HELBURUAK ETA IRISPENA

Proiektuaren helburu nagusia Matlab bidezko optimizazio kode batetik abiatu eta egituraren ingeradak lortzea da. Ekintza hau gauzatzeko lehenengo optimizazio topologiarekin egitura lortuko da. Egitura honen ingeradak nahiko irregularrak direnez, kurba hauek leunagoak izatea lortu behar da. Aurrez aipatutako ingerada edo inguruak azken kurba leun hauek dira. Honela pieza lortzeko fabrikazioaz baliatuz, 3 dimentsiotan egitura inprimatzeko bidea jarraituko da.

Helburu nagusia aipatu den arren proiektua gauzatu bitartean hainbat azpi-helburu daude. Hemen beheran azaltzen dira hauek:

- Hasiera batean Excel eta Matlab, optimizazio topologikoa gauzatu den programa, bidezko lotura egokia ezartzea.
- Excel-en grafiko egokia lortzea ordenagailu bidez Matlab softwareko erantzunak erabiliz. Honela, esan bezala ingeraden kurba suabeagoak lortzea.
- Azken erantzun hauen bitartez, CAD artxibo bat sortuko da. CAD artxibo hau 3 dimentsiotako piezak lortzeko programak erabiliz gauzatuko delarik.
- CAD artxibo honetaz baliatuz aurrez aipatu den piezaren fabrikazioa praktikan jarriz, 3D inprimagailu bidez solidoa lortu nahi da.
- Amaitzeko, Matlab formatutik CAD formaturako aldaketa ahalik eta automatikoena lortzea izango da helburu. Hasiera batean pausuak banan-banan egingo dira lortu nahi dena posiblea dela egiaztatzeko. Baina, automatikoki egitea aurrerapen handia denez hau izango da lortu nahiko dena etorkizun batean izango duen erabilgarritasunari begira.

Proiektuaren irispenari dagokionez Matlab kodetik, ‘ *A 99 line topology optimization code written in Matlab* ’ deitzen denetik, abiatzen da. Programa hau piezari indarra aplikatzean, zenbait parametroren menpean, tarteko dentsitateak hartuko dituzten balioen arabera piezaren optimizazio topologikoa gauzatuko da.

Honetaz baliatuz, lanaren amaierara helduz CAD artxiboa lortzean datza, beti ere ahalik eta modu automatikoenean. Azkenik, pieza honen datuak 3D inprimagailura bidali eta pieza modu solidoan lortzean proiektuari bukaera emango zaio.

4 LANAK DAKARTZAN ONURAK

Atal hau lan honek dakartzan onuren ingurukoa da, izan ere, onura hauek dezente dira. Proiektuan optimizazio topologiko bidez lortutako egitura bat eraikitzean datza, hau da, egitura idealena eraikitzean. Horregatik esan daiteke etorkizuneko fabrikazio bihurtuko dela. Gainera, gaur egun fabrikazio gehigarriak emandako bultzadarekin duen lotura ezagutuz, biak bat eginik pieza konplexuen etorkizuneko eraikuntza honetan oinarrituko dela esan daiteke.

Lanaren onurak bi atal nagusitan banatuko dira, alde batetik onura teknikoak, eta bestetik ekonomikoak.

4.1 LANAREN ONURA TEKNIKOAK

Onurak dakartzaten alderdi teknikoak hainbat dira, baina esanguratsuenak aipatuko dira hemen. Horregatik hiru talde nagusi banatuko dira, hauek optimoa izatea, automatikoa izatea eta 3D bidezko inpresioaren ingurukoak izango dira.

Hasteko atal optimoari begira, honen abantaila nagusia diseinuaren hobekuntza nabarmena da. Honela diseinua hobea izatean ezaugarri mekanikoak ere hobeak izatea eragiten du. Ezaugarri mekaniko hauek besteak beste erresistentzia eta zurruntasun mekanikoa izango dira.

Alderdi automatikoak esan nahi duena da, pieza bat zerotik hasita automatikoki egitea. Beraz, kalitate goreneko pieza optimoak egiteaz gain fabrikazio tradizionalean baina abiadura handiagoan egiten badira egundoko onura dela esan daiteke.

Azkenik 3 dimentsiotako inpresioari begira nabarmendu behar da edozein forma lortzeko gai dela. Gainera, inprimagailuaren dimentsioak kontuan izanda hainbat tamainatako egitura edo piezak lortu daitezke. Optimizazio topologiko bidez lortutako piezak izanda ere, hau da, forma arraroak inprimatu daitezke.

4.2 LANAREN ONURA EKONOMIKOAK

Onura ekonomio eta teknikoen arteko lotura ezartzen da, batak besteari zuzenean eragiten diolako. Horregatik hemen ere, hiru taldetan banatuko da materiala, denbora eta 3D inpresioa izanik.

Materialaren atalarekin hasiz, esan behar da pieza geroz eta optimoagoa izan material aurreztea handiagoa izango dela hau eraikitzeko. Beraz, material gutxiago erabiltzeak azkeneko prezioaren murrizketa proportzionala ekarriko du.

Alderdi automatikoarekin zurrunki lotuta dagoen denbora aurreztearen atalarekin jarraituko da. Honela, aurretik aipatu bezala eraikitzeko prozesua azkarragoa bada esan nahi du denbora gutxiago beharko dela ekoizkina lortzeko. Hori dela eta, produkzio denbora laburrak prezioak txikitzea eragingo du.

Amaitzeko 3D inpresioaren atalean aipatzekoa da gaur egun garesti xamarra dela metodo honen bidez egiturak inprimatzea. Hala ere, etengabeko garapean dagoen arlo hau oso azkar doanez, denbora gutxiren buruan prezio hauek jaitsiz joango dira.

5 OPTIMIZAZIO TOPOLOGIKOAREN PROBLEMA

Optimizazio topologikoko problema batean hasierako domeinu batetik abiatzen da, orokorrean errektangeluarra dena. Domeinu hau hartu eta diskretizatu egiten da topologia optimoa lortzeko karga baldintza eta mugarri (euskarri) batzuen funtziopean. Optimizazio topologikoa abiapuntutzat erabiltzen da diseinu berrientzat, izan ere, egiturek karga batzuen pean izango duten funtzionamendu optimorako izan behar duenaren ideia orokorra aurkezten dutelako. Nahiz eta ondoren azken karga hauen aldaketa posibleak kontuan izan behar diren.

Kasu honetan egitura jarraituen optimizazio topologikoa burutuko da, non domeinua elementu finitu bidez banatuko den. Elementu hauek materialaren zati txikien banaketa aurkezten dute. Elementu finitu bakoitza iteratiboki aztertuko da materialaren banaketa optimoa lortu arte.

5.1 OPTIMIZAZIO TOPOLOGIA EZARTZEKO PROZEDURA

Optimizazio topologikoaren analisia egiteko honako pausu jakin hauek jarraitu behar dira:

- Problemaren deskribapena: hasierako domeinua, jasango dituen kargen baldintzak, murrizketak eta diseinuan parte hartzen duten gainontzeko aldagaiak zehaztean datza.
- Elementu finitu bidezko domeinuaren diskretizazioa.
- Optimizazio topologikoko algoritmoaren inplementazioa, honek elementu finituak iteratiboki aztertzen ditu behar denean materia deuseztatuz azken soluziora heldu arte. Kontuan izan behar da lortzen diren erantzunak matematikoak direla, horregatik gertatu daiteke material errealei aplikatzerakoan ilogikoak izatea.
- Lortutako emaitzen interpretazio eta egiaztapena, eta segidan ondorioen lorpena eta objektuaren fabrikazioa.

5.2 PROBLEMAREN FORMULAZIO OROKORRA

Optimizazio topologikoko prozesuan problema matematikoa ebazteko 3 taldetan banatzen diren elementuak egon behar dira:

1. **Helburu den funtzioa**

Problemaren aldagaien funtzio gisa definitzen da sistemaren eraginkortasuna. Orokorrean, aldagai hauen balioa minimizatzea bilatzen da.

2. **Murrizketak**

Murrizketak problemari mugak jartzean datza, non erantzun posibleen artean aukerak txikitzen diren. Murrizketa hauek esplizitu edo inplizituak izan daitezke. Esplizituak zuzenean eragiten dute, aldagai bati edo batzuei baldintzak jarriz. Inplizituak berriz aldagaien menpe dauden kalkulatutako magnitudeei eragiten diete. Oro har, murrizketa esplizituak ebazteko errazagoak dira.

Murrizketak banatzeko beste modu bat berdintasun eta desberdintasun bidezkoa da. Berdintasunezkoak egituraren portaerari dagozkion legeekin erlazionatuta daude, esaterako elementu finkoak, materialen portaera legeak, eta abar. Desberdintasunezko murrizketak inposatutako mugekin erlazionatuta daude, adibidez, desplazamendu edo tentsio maximoak, bete dezakeen bolumen maximoa, eta abar.

3. **Problemaren aldagai eta parametroak**

Aldagaiak ebatzi beharreko problemaren ezezagunak dira. Hauek diseinu prozesuan zehar aldatzen doaz. Honela banatu daitezke:

- Sekzioaren propietateak, inertzia momentua edo azalera besteak beste.
- Egitura elementuaren geometria: lodiera, luzera, altuera, etab.
- Egitura elementuaren topologia: materialaren dentsitatea, perimetro totala, etab.

Sistemaren parametroak mugarri eta helburu funtzioaren aldagaiak erlazionatzen dituen balio ezagunak dira. Prozesu guztian zehar konstante dirauten balioak dira.

Orokorki problema azpikoan oinarritzen da:

- *f(x)* minimizatzea

- Baldintzak $g_j(x) \leq 0$

  $h_k(x) = 0$

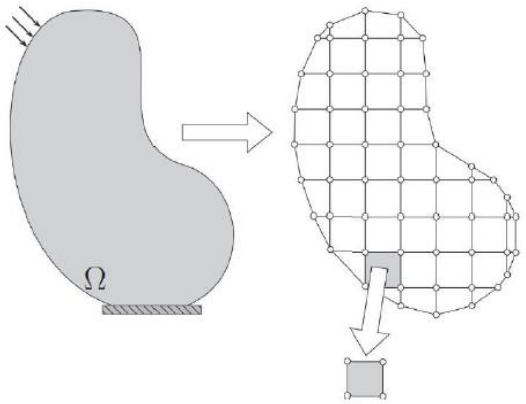
- $x_i^{min} \leq x_i \leq x_i^{max}$ izanik

Non:

$f(x)$:	Helburu funtzioa
x:	Diseinu aldagaiak dituen bektorea
x_i	Aldagai bektorearen elementu bakoitza
x_i^{min}, x_i^{max}	Elementuen diseinu mugak
$g_j(x)$	Desberdintasun murrizketak
$h_k(x)$	Berdintasun murrizketak

5.3 ELEMENTU FINITU BIDEZKO PROBLEMA ELASTIKOA

Elementu finituen metodoa domeinu jakin bat nodo bidez lotzen diren azpi-domeinutan (elementu finituak) diskretizatzean datza, hurrengo irudian (Irudia 3) ikusi daitekeen bezala. Elementu finitu bidezko soluzioa ezaguna da eta helburu funtzioa inguratzen duten 4 nodoei aplikatuz lortzen da. Honela ekuazio diferentzialen problema elementu finitu bidezko berehalako ebazpenera murrizten da.



Irudia 3. Elementu finitu bidezko diskretizazioa.

Optimizazio topologikoan, helburu funtzio ezagunena zurruntasun maximoarena da non malgutasuna minimizatzearen berdina den, beti ere erabili daitekeen bolumen kantitate maximoaren baldintza inposatzea da ohikoena.

5.4 PROBLEMAREN DESKRIBAPENA

Erabiltzen den Matlab kodearen arabera lan honen optimizazio topologikoaren problema azalduko da atal honetan. Aipatzekoa da, kode hau errazagoa izateko zenbait sinplifikazio dauzkala. Hasteko diseinuko domeinua errektangeluarra da eta elementu finitu karratuen bidez diskretizatzen da. Honela, elementu eta nodoen zenbaketa errazten da, hau goiko ezkerreko ertzetik abiatzen delarik. Egituraren ratioa horizontal eta bertikalean kokatutako elementuen ratioak emandakoa da.

Optimizazio Topologikoaren problema azpiko adierazpenean oinarritzen da, helburua onarpen hau minimizatzeko honela idaztea da,

$$\left.\begin{aligned} \min_{\mathbf{x}}: \quad & c(\mathbf{x}) = \mathbf{U}^T\mathbf{K}\mathbf{U} = \sum_{e=1}^{N} (x_e)^p \, \mathbf{u}_e^T \, \mathbf{k}_0 \, \mathbf{u}_e \\ \text{subject to}: \quad & \frac{V(\mathbf{x})}{V_0} = f \\ : \quad & \mathbf{K}\mathbf{U} = \mathbf{F} \\ : \quad & \mathbf{0} < \mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{1} \end{aligned}\right\} \quad \textbf{(1)}$$

non **U** eta **F** desplazamendu eta indar bektore globalak diren, hurrenez hurren, **K** zurruntasun matrize globala den, $\mathbf{u_e}$ eta $\mathbf{k_e}$ elementuen desplazamendu bektore eta zurruntasun matrizea diren, hurrenez hurren, **x** diseinu aldagai bektorea den, $\mathbf{x_{min}}$ dentsitate erlatibo minimoa dutenen bektorea den (zero gabea singularitateak saihesteko), **N** (= `nelx x nely`) diskretizatutako diseinu domeinuaren elementu kopurua den, **p** penalizazioa den (normalean p = 3), **V(x)** eta $\mathbf{V_0}$ materialaren bolumena eta diseinuko domeinuaren bolumena diren, hurrenez hurren eta **f** (`volfrac`) ezarritako bolumen frakzioa den.

Optimizazio problema hau **(1) adierazpenaren** aukera desberdinen bitartez ebatzi ahalko litzateke. Adibidez, aurretik aipatutako optimizazio irizpide (OC), programazio lineal sekuentzial (PLS) edo asintota mugikorren metodoa (MMA) erabiliz beste batzuen artean. Sinpletasuna bermatuz, OC-metodoa erabili da.

Jarraian Bendsøe (1995) irakasleak eguneratutako plan heuristikoa adierazten da diseinu aldagaientzat,

$$x_e^{\text{new}} =$$

$$\begin{cases} \max(x_{\min}, x_e - m) \\ \quad \text{if} \quad x_e B_e^{\eta} \leq \max(x_{\min}, x_e - m)\,, \\ x_e B_e^{\eta} \\ \quad \text{if} \quad \max(x_{\min}, x_e - m) < x_e B_e^{\eta} < \min(1, x_e + m) \\ \min(1, x_e + m) \\ \quad \text{if} \quad \min(1, x_e + m) \leq x_e B_e^{\eta}\,, \end{cases} \qquad \textbf{(2)}$$

non **m** (move) mugimendu limite positiboa den, **η** (=1/2) koefiziente numeriko bat den eta $\mathbf{B_e}$ baldintza optimoa bilatzekoa den

$$B_e = \frac{-\dfrac{\partial c}{\partial x_e}}{\lambda \dfrac{\partial V}{\partial x_e}} \qquad \textbf{(3)}$$

non **λ** biderkatzaile Lagrangiarra den eta 'bi-sectioning' edo erdibiketa algoritmo baten bidez lortu daitekeen.

Funtzio nagusiaren sentikortasuna azpiko adierazpenaren bitartez aurkitu daiteke.

$$\frac{\partial c}{\partial x_e} = -p(x_e)^{p-1}\ \mathbf{u}_e^T\ \mathbf{k}_0\ \mathbf{u}_e \qquad \textbf{(4)}$$

Optimizazio topologikoko problemak **(1) adierazpenaren** erantzun bat izan dezan, zenbait murrizpen aplikatu behar dira. Matlab kode honetan filtro bidezko teknika bat (Sigmund 1994, 1997) erabili da. Enfasia egin behar da ez baita frogatu filtro honekin emaitza existituko denik, baina hainbat aplikazioren bitartez frogatu da sare independentearen diseinua egiten duela praktikan.

Sare independentzi filtroak elementuen sentikortasunak aldatuz lan egiten du, honela:

$$\widehat{\frac{\partial c}{\partial x_e}} = \frac{1}{x_e \sum_{f=1}^{N} \hat{H}_f} \sum_{f=1}^{N} \hat{H}_f\ x_f\ \frac{\partial c}{\partial x_f} \qquad \textbf{(5)}$$

Konboluzio operadorea (pisu faktorea) $\hat{\mathbf{H}}_\mathbf{f}$ honela idazten da

$$\hat{H}_f = r_{\min} - \operatorname{dist}(e, f)\,, \quad \{f \in N \mid \operatorname{dist}(e, f) \leq r_{\min}\}, \quad e = 1, \ldots, N \qquad \textbf{(6)}$$

non operadorearen **dist (e, f)** e eta f elementuen zentroen arteko distantzia den. Konboluzio operadorea $\hat{H}_f$ filtratutako azaleratik kanpo zero da, eta linealki txikitzen da f elementuaren distantziaren arabera. Sentikortasun originalak **(4)** erabili beharrean, sentikortasun aldatuak **(5)** erabiltzen dira optimizazio adierazpen gaurkotuan **(3)** [6].

6 MATLAB KODEAREN INPLEMENTAZIOA

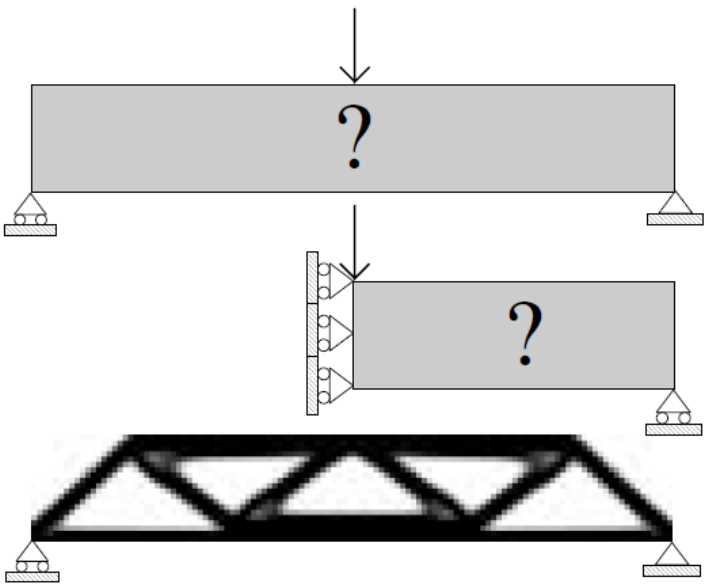
Proiektu honetan zehar erabilitako Matlab kodea optimizazio topologikoaren kode estandar bat dela esan daiteke, hau I . eranskinean erakusten dena izanik. Kodearen izena aurretik aipatu den moduan *‘ A 99 line topology optimization code written in Matlab ’* da eta egitura optimoa lortzeko erabili da. Programa nagusia funtzio bat da, Matlab-eko komando leihotik deitzen dena honakoa idatzita:

```
top(nelx,nely,volfrac,penal,rmin)
```

non `nelx` eta `nely` elementu kopurua diren domeinuaren ardatz horizontal eta bertikalean, hurrenez hurren, `volfrac` betea izango den bolumen frakzioa den, `penal` penalizazio faktorea den eta `rmin` filtroaren erradioa den (elementuaren sekzioarekin zatituta).

Gainontzeko aldagaiak ingurumen baldintzekin batera Matlab kodean bertan definitzen dira eta behar ezkero aldatu daitezke. Iterazio bakoitzean optimizazio topologiaren begiztan, kodean momentuko dentsitate banaketaren irudia sortzen du. Ondorengo irudiak (Irudia 4) erakusten du Matlab kodea azpiko sarrera hauekin exekutatzean lortzen den dentsitate banaketa

```
top(60,20,0.5,3.0,1.5)
```



Irudia 4. Optimizazio topologikoa Matlab kodearen bitartez lortua. Gainean: domeinuaren diseinu totala. Erdian: diseinuko domeinua simetria aplikatuta. Behean: optimizazio topologikoaren erantzuna (bi erdiak).

Lehenetsitako ingurune baldintzak gaineko irudiko habe erdiarekin bat egiten dutenak dira. Egitura simetrikoa kontsideratuz, karga ezkerreko gaineko ertzean aplikatzen da eta ingurune baldintzak simetrikoak dira ezkerreko ertzean zehar. Egitura horizontalean finkatuta dago eskuineko azpiko ertzean duen euskarriaren bidez.

Programa osoa zenbait azpi-atalez osatzen da eta hauek jarraian azaltzen dira.

6.1 Programa nagusia (1-36 lerroak)

Programa nagusia (1-36 lerroak) diseinuko domeinuan materia banatuz (4.lerroa) hasten da. Beste hasieratze batzuen ostean, begizta nagusian sartzen da elementu finituen (12.lerroa) azpirrutina ebazteko non `U` desplazamendu bektorea bueltatzen duen. Elementuen zurruntasun matrizea material solidoentzat berdina denez elementu guztientzat, nahikoa da elementuen zurruntasun matrizearen azpirrutinari behin deitzea (14.lerroa). Jarraian, helburu funtzioa eta sentikortasuna elementu guztietan zehazten dituen zikloan (`loop`) sartzen da (16-24 lerroak). `n1` eta `n2` aldagaiak gaineko ezker eta eskuin elementuaren nodo zenbakiak nodo zenbaki globaletan adierazten dira, hauek desplazamendu globalaren bektoretik `U` elementuen desplazamendu bektorea U_e ateratzeko erabiltzen direlarik. Sentikortasun analisiaren jarraian saretze independentearen filtroa (26.lerroa) dator, baita optimagarritasun irizpidearen optimizazioa (28.lerroa) ere. Orain arte onartutako aldagaiek beste parametro batzuekin batera pantailaratzen dira 30-33 lerroen artean eta lortutako dentsitate banaketa irudikatzen da (35.lerroa). Ziklo nagusiari amaiera ematen zaio diseinu aldagaien aldaketa (`change` 30.lerroan izendatua) ehuneko 1 baino txikiagoa baldin bada. Gainontzean gaineko pausuak berriz ere errepikatzen dira.

6.2 Optimagarritasun irizpidean oinarritutako optimizazioa (37-48 lerroak)

Eguneratutako diseinu aldagaiak optimizazioaren bidez bilatzen dira (37-48 lerroak). Jakinik materialaren bolumena `(sum(sum(xnew))` Lagrange biderkatzearen (`lag`) funtzio monotonoki beherakoa dela, Lagrangian biderkatzailearekin bat egiten duen bolumenaren muga balioa bi-sectioning edo erdibiketa algoritmoaren bidez aurkitu daiteke (40-48 lerroak). Bi-sectioning algoritmoa hasieratzeko Lagrangian biderkatzailearen beherengo `l1` eta gorengo `l2` mugak susmatzen dira. Muga hauen

(`l1-l2`) tartea Lagrangian biderkatzailea erdira murriztu eta hauen neurria konbergentzi irizpidea baino txikiagoa izan arte errepikatzen da (40.lerroa).

6.3 Saretze independentearen filtroa (49-64 lerroak)

49-64 lerroek Matlab bidezko inplementazioaren saretze independentearen filtroa erakusten dute. Aipatzekoa da diseinuko domeinuaren elementu guztiak ez direla bilatu `rmin` erradioaren barnean kokatzeko, baizik eta, `round(rmin)` sekzioko luzera duen elementu karratua kontsideratutako elementuaren inguruan kokatzeko. Aldagai honen, `rmin`, balioa bat baino txikiagoa izateko hautatzen bada, filtroaren sentikortasuna originalaren berdina izango da ez aktibatzea eraginez.

6.4 Elementu finituen kodea (65-99 lerroak)

Elementu finituen kodea 65-99 lerroen bitartean dago idatzita. Esan behar da, soluzio bilatzaileak Matlab softwarearen `sparse` aukera erabiltzen duela, matrizearen elementuak banan-banan bistaratzen dituena. Zurruntasun matrize globala elementu guztiak zeharkatzen dituen ziklo baten bidez sortzen da (70-77 lerroak). Programa nagusian bezalaxe, `n1` eta `n2` aldagaiak gaineko ezker eta eskuin elementuaren nodo zenbakia adierazten du nodo zenbaki globaletan, eta elementuaren zurruntasun matrizea zurruntasun matrize globalaren kokapen aproposean jartzeko erabiltzen da.

Aurretik aipatu den moduan, nodo eta elementuak ezkerretik eskuinerako zutabeen bidez zenbatzen dira. Bestalde, nodo bakoitzak bi askatasun gradu dauzka (horizontala eta bertikala). `F(2,1)=-1` komandoak (79.lerroa) bertikalean unitate bateko beheranzko indar bat goiko ezkerreko ertzean aplikatzen dela esan nahi du.

Euskarriak askatasun gradu finkoak ezabatuz ekuazio linealen bitartez ezarritakoak dira. Matlab bidez ekintza hau dotoreki egin daiteke 84 lerroan ikusi daitekeen bezala.

```
U(freedofs,:) = K(freedofs,freedofs)\F(freedofs,:);
```

non `freedofs` aldagaiak derrigorrezkoak ez diren askatasun graduak adierazten dituen. Normalean errazagoa da finko dauden askatasun graduak (`fixeddofs`) definitzea, honela `freedofs` aldagaiak automatikoki bilatzen dira Matlab operadorearen `setdiff` komandoa erabiliz. Azken komando honek askatasun

graduak bilatzen ditu askatasun gradu guztien eta askatasun gradu finkoen diferentzi gisa (82.lerroa).

Elementuaren zurruntasun matrizea (8x8 tamainakoa) 86-99 lerroetan zehar kalkulatzen da. Matrizeko elementu finituak 4 nodoko elementu bi lineal karratuak dira. Young-en modulua `E` eta Poisson-en ratioa `nu` 88 eta 89 lerroetan aldatu daitezke [6].

7 OPTIMIZAZIO TOPOLOGIKOAREN INTERPRETAZIOA

Aurreko atalean azaldutako Matlab kodearen bitartez lortzen den optimizazio topologikoaren interpretazioa bi modutara burutu da. Alde batetik optimizazio topologikoaren bitartez lortzen den x matrizea, dentsitate erlatiboen balioak biltzen dituen matrizea, hartzen da zuzenean. Eta bestetik, Matlab bidez azpi-programa bat sortzen da x matrizeak dituen dentsitate erlatiboak ezabatzeko.

7.1 X MATRIZE BIDEZKO INTERPRETAZIOA

Hasiera batean, kodea exekutatzean lortzen den x matrizea, hau da, erantzuna, hartzen da aurretik aipatu den bezalaxe. Hau lortu ostean, excel orri batera bidaltzen da matrizea hurrengo atalean azalduko den prozedura jarraituz.

Behin excel orrian gaudela, 3 dimentsiotako grafikoa marrazten da lortutako egitura optimoaren itxura berriz ere ikusteko, lehenengo bistaratzea Matlab softwarean egiten baita (Irudia 4). Marraztutako grafiko honetan, goitiko bistan kokatuz balio tarte ezberdinak dituzten definizio eremuak ikusten dira. Horregatik, definizio eremu hauek goiko bistaren planoaren ardatz perpendikularrean barrurantz neurri desberdinetara ebaketak eginez sailkatuko dira, erantzun egokiena baliogarritzat hartuz. Izan ere, definizio eremu batetik besterako dagoen desberdintasuna ingeraden irregulartasunari dagokiona da, hots, behar baino koska gehiegi izatea adibidez eta hau bilatzen denaren aurkakoa da.

7.2 MATLAB BIDEZKO AZPI-PROGRAMAREN BITARTEZKO INTERPRETAZIOA

Matlab bidez egindako azpi-programak x matrizearen dentsitate erlatiboak ezabatu eta materiala dagoen eta ez dagoen guneen bereizketa egitea du helburu. X matrizearen dentsitate erlatiboak optimizazio topologikoak ematen duen erantzuna da, eta azpi-programa hau exekutatuz aurretik zeuden gune grisak ezabatuko dira esan bezala.

Hasteko komando leihoan `puntuak(x)` idatziko da azpi-programari deitu eta bera bakarrik exekutatzeko. Parentesi artean dagoen x aurretik aipatutako izen bereko matrizea da.

Azpi-programarekin hasteko lehenengo 8 lerroetan x matrizeari lerro eta zutabe bat gehitzen zaio (5 eta 6.lerroak), biak 1 balioko bektoreak direlarik. Hau da, lerro gehigarria 1 x (lerroak + 1) neurriko bektorea da eta zutabe gehigarriaren bektorea (zutabeak + 1)x1-ekoa. 7.lerroko `n` balioa puntuz sortutako matrizean posizioak kokatzeko kontagailua da.

```
%% DENTSITATE ERLATIBOAK EZABATZEKO AZPI-PROGRAMA %%
function pto = puntuak (x)
%% Matrizeari zutabe eta lerro bat gehituz konparaketa egin
[fl,cl]=size(x);
x(fl+1,:)=ones;
x(:,cl+1)=ones;
    n=0;
[f,c]=size(x);
```

Ondorengo lerroetan x matrizea zeharkatzen da ondorengo hiru baldintzak aplikatuz. Lehenengo matrizearen posizio jakin bateko balioa 0.8-ko dentsitate erlatiboaren berdina edo handiagoa bada, matrizearen posizio hau A izeneko matrizean gordeko da, hots, posizioak gordetzen dituen matrizean. Bestalde, matrizearen posizio zehatzean dagoen balioa 0.3 baino txikiagoa baldin bada, dentsitate erlatiboa zerora borobilduko da. Balio nulu hau ez da posizioen matrizean sartuko materialik gabeko puntua delako. Hurrengo bi baldintzetan gaineko eta alboko posizioetako dentsitate erlatiboak konparatuko dira. Hauen eginkizuna, aztertzen ari den posizioko dentsitatea handiagoa bada bere posizioa aurretik aipatutako A matrizean gordetzea da. Kasu honetan konparaketa bi balioen artean egiten da handipenik erabili gabe edo 1 balioko handipena dagoela esan daiteke. Hala ere, handipen handiagoko kasuak ere aztertzen dira balio egokiena aukeratu arte. Eta handipenaz gain, balio desberdinak diren kasua ere aztertzen da `~=` ikurra erabiliz desberdintza adierazteko, kasu hau guztietan kontserbakorrena da. Lehen ikusitako `n` parametroa hemen ikusten da nola balio bat handituz joango den posizio bat gordetzen den bakoitzean.

```
%% Matrizea zeharkatzea
for i=1:c-1 ;
  for j=1:f-1 ;
    if (x(j,i) > 0.8)
    posiziox=i;
    posizioy=j;
    n=n+1;
    A(n,1)= posiziox;
    A(n,2)= posizioy;
    elseif (x(j,i) < 0.3)
    x(j,i)=0;
    elseif (x(j,i) > (1*x(j+1,i)))
    posiziox=i;
    posizioy=j;
    n=n+1;
    A(n,1)= posiziox;
    A(n,2)= posizioy;
    elseif   (x(j,i) > (1*x(j,i+1)))
    posiziox=i;
    posizioy=j;
    n=n+1;
    A(n,1)= posiziox;
    A(n,2)= posizioy;
    end
  end
end
```

Azkeneko zatian puntuen posizioak gorde dituen matrizea Matlab softwareko komandoen leihoan bistaratuko da 34.lerroan ikusten den modura. Bukatzeko 35.lerroko adierazpena hurrengo atalean azalduko den Matlab eta excel bidezko lotura da, beraz excel-ean hainbat puntuz osatutako taula bat agertuko da. Taula horretatik abiatuz dispertsio motako grafiko bat egingo da x eta y ardatzak dituena eta puntuen koordenatuak marrazten dituena [9].

```
%% Posizio puntuen matrizea irudikatu
pto = A
status = xlswrite('puntuak.xlsx',pto,'Hoja1','B2');
end
```

8 MATLAB ETA EXCEL BIDEZKO LOTURA

Bi programa hauen arteko lotura oso garrantzitsua da proiektu honetan, izan ere, Matlab bidez lortutako optimizazio topologikoa, helburu den lez, beste formatu batzuetara bilakatzeko aukera ematen duelako. Ekintza hau burutzeko komando sinple bat besterik ez da behar. Hala ere, komando hau erabili ahal izateko `io` izeneko paketea deskargatu behar da Matlab softwarean bertan. Deskarga hau burutzeko komandoen leihoan azpiko hau idatzi behar da.

```
>> pkg load io
>> pkg list
```

Lehenengo komandoak `load` hitzak adierazten duen bezala paketea deskargatzeko balio du, eta hurrengoak ordea pakete guztien zerrenda bistaratzen du. Zerrenda honetan aipatutako paketea aktibatu dela jakiteko `io*` moduan agertuko da komando leihoan.

Paketea aktibatuta dagoenean, azpiko komandoa idaztean excel orri bat zabalduko da nahi den taula edo balioekin. Komandoa honako hau da:

```
status = xlswrite('izena.xlsx',balioa,'Hoja1','B2');
```

non `izena.xlsx` excel artxiboak edukiko duen izena den, balioa excel-era inportatuko den taula den, `Hoja1` eta `B2` nahi den taula edo balioa excel-ean bertan zein orri eta posiziotan kokatuko den adierazten duen eta `status` aldagaiak 1 balioa bueltatuko duen funtzioa bete dela adierazteko [9].

9 CAD SOFTWAREAREN INPLEMENTAZIOA

Optimizazio topologikoaren interpretazioa bi bide posibleetatik egin den moduan, CAD softwarea hiru eratara aplikatuko da aurretik aukeratutako bidearen arabera. Hala ere, erabiliko diren softwareak C++, Autocad eta Solid Edge izango dira, azkeneko hau hiru bideetan erabiliko delarik. Hiru era hauetatik lehenengoa, x matrizea zuzenean optimizazio topologikoaren emaitzatik hartzen duena da, eta beste biak, Matlab bidezko azpi-programari jarraipena ematen diotenak izango dira.

9.1 X MATRIZEAREN EXCEL-etik ABIATUZ

X matrize honen excel-eko grafikotik abiatuta Solid Edge bitartez egitura sortuko da atal honetan. Lehenik eta behin, excel-eko grafikoan balio tarte egokiena aukeratzen denean, grafiko hori irudi moduan gordeko da.

Behin lehenengo pausu hau eginda, Solid Edge zabalduko da, ‘ *ISO métrico Pieza* ’ artxibo berri bat zabalduz, gaztelerako bertsioa baldin bada. Artxibo berria zabaltzen denean arratoiaren eskuineko botoiari zapalduz ‘ *Pasar a Ordenado* ’ emango zaio eta aldi berean sarrerako orrian ‘ Boceto ’ aukeran klikatuko da. Segidan, bozeto edo zirriborroa egiteko x-y planoa aukeratuko da, matrizearen grafikoan erabilitako berdina. Hurrengo pausuan excel-eko grafikoaren argazkia Solid Edge softwarera inportatuko da, horretarako pausu sekuentzia hau jarraituko da ‘ Herramientas - Insertar – Imagen ’.

CAD software honetan egin nahi den piezaren irudia dagoenean honen gainean marraztea besterik ez da geratzen. Honela lerro edo spline itxurako kurbak erabiliko dira kasu bakoitzean hoberen egokitzen dena hartuz. Kontuz ibili behar da lerro guztiak itxiak izan behar direlako. Beraz, hau egin ostean zirriborrotik irtengo da eta pieza solido bihurtuko da lodiera bat emanez. Lodiera emateko lehenengo ‘ Cerrar Boceto ’ aukera klikatuko da, eta ondoren lodiera lortzeko ‘ Extrudir ’ aukerari emango zaio. Azkeneko hau klikatzean ‘ Seleccionar Cadena ’ aukeratuko da dagozkion lerro itxiak aukeratuz, amaitzeko nahi den lodiera emango zaio piezari. Esan behar da lortzen den pieza berez lortuko zenaren erdia dela, horretarako simetria aplikatzea besterik ez da egin behar. Gainera kasu honetan ez dago biribilketak egiteko beharrik erabilitako kurbak nahiko leunak direlako.

Amaitzeko lortu den pieza hau .stl artxibo bat bezala gordeko da eta hiru dimentsioetan inprimatzeko prest geratuko da. Modu honetan gordetzeko ‘ Guardar como ’ ataletik egin behar da, automatikoki egiten bada .par moduan gordetzen delako [10].

9.2 MATLAB BIDEZKO AZPI-PROGRAMATIK ABIATUZ

Bigarren eta hirugarren kasu hauetan Matlab bidez garatutako azpi-programatik abiatzen da, hau da, aurretik azaldu bezala excel-eko puntuz osatutako taulatik. Hasteko bietan excel-eko orrian bertan ‘ Concatenar ’ komandoa erabiliz puntuak ‘ x,y ’ koordenatu moduan jartzen dira, alde batetik Autocad eta bestetik C++ softwareek irakurri ditzaten.

9.2.1 Puntuak zuzenean erabiliz

CAD softwareekin hasteko lehenik eta behin Autocad zabalduko da orrialde berri batekin, non orri hau ISO norma errespetatzen duena aukeratzen delarik. Kontuz ibili behar da Autocad 3 edo 2 dimentsiotan egon daitekeelako eta puntuak soilik bi koordenatu dituzte, beraz artxiboa 2 dimentsiotan dagoela ziurtatu behar da. Programa honetan puntuak automatikoki sartzen dira ‘ PUNTO ’ izeneko komandoa erabiliz. Azken hau erabiltzeko excel-ean dauden puntuak kopiatu egingo dira lehendabizi, eta segidan komando lerroan ‘ PUNTO ’ idatzi ostean kopiatuko dira. Hau egin ostean, ‘ *Enter* ’ tekla sakatuz puntu guztiak pantailan ikusiko dira. Autocad erabiltzen bukatzeko sortutako artxiboa gordeko da lehenetsita dagoen formatuan, hau da, .dwg formatuan. Izan ere, ondoren erabiliko den Solid Edge softwarea formatu hau irakurtzeko gai da [11].

Autocad itxi ostean Solid Edge zabalduko da, prozesuarekin jarraitzeko. Bi programa hauek erabiltzearen arrazoi nagusia da Autocad softwareak puntu kantitate handiak automatikoki sartzen dituela programan, baina Solid Edge-k ordea banan-banan sartzea behartzen du. Bestalde, Solid Edge erabiltzeak geroago inprimatzeko erraztasuna ematen du.

Beraz, azken programa hau zabalik dagoela, aurretik itxi den Autocad-eko artxiboa zabalduko da ‘ *iso metric part.par* ’ aukera hautatuz, ISO norma hau baita aurretik erabili dena. Puntuen artxiboa zabaltzen denean aurreko bidean erabilitako pausu

batzuk errepikatzen dira. Hasteko eskuineko botoia sakatu eta ‘ *Pasar a Ordenado* ’ emango da. Jarraian ‘ Boceto ’ aukeraren gainean klikatuko da eta zirriborroa egiteko x-y planoa aukeratu, izan ere hau da puntuak kokatzen diren planoa.

Dagoeneko puntuak lerro edo spline itxurako kurben bitartez elkartzea besterik ez da geratzen, modu honetan inguru edo ingeradak sortzen direlarik. Aurretik aipatu bezala oraingoan sortzen diren ingeradak ere itxiak izan behar dira. Honela inprimatu aurreko azkeneko pausua geratzen da, beste ibilbidearenaren berdina dena. Hots, piezari lodiera emango zaio lehenik eta behin ‘ Cerrar Boceto ’ aukera klikatuz. Lodiera lortzeko aurretik aipatu bezala ‘ Extrudir ’ hautatuko da, beti ere ‘ Seleccionar Cadena ’ aukera erabiliz. Horregatik kateak edo, beste era batera esanda, ingeradak aukeratuko dira. Eta bukatzeko, lodieraren neurria zehazten da solidoa lortzeko. Aipatzekoa da bestea kasuan bezala hau ere berezko piezaren erdia dela eta simetria aplikatuz pieza oso lortzen dela. X matrizearen kontrara kasu honetan bai egin behar dira biribilketak zenbait ertzetan. Biribiltzeko ekintza hau ‘ Redondear - Todos los acuerdos ’ eta biribiltzeko erradioa 1mm-koa aukeratuz automatikoki egiten du softwareak. Biribilketa hauekin ertz zorrotzak kentzean tentsio kontzentrazio eremuak aldi berean gutxitzen dira, izan ere, ertz zorrotz eta erradio aldaketa bortitzak tentsio kontzentrazioen ohiko guneak dira.

Aurreko ibilbidearen amaiera bera jarraituz, artxibo hau .stl moduan gordeko da inprimagailuak irakurri eta solidoa sortu dezan [10].

9.2.2 Puntuak segmentuen bidez lotzeko programatuz

Puntuak segmentuen bidez lotzeko C++ bidez garatutako programa bat erabili da, zehazki II . eranskinean erakusten dena. Programa hau deskribatzeko zenbait zatitan banatzen da, alde batetik programa nagusia dago azpi-programa desberdinei erreferentzia egiten diena, eta bestetik azpi-programa hauen garapena.

Programa nagusiarekin hasiz, hau 226-239 lerro tartean kokatzen da eta ikusi daitekeen bezala bere eginkizuna azpi-programei deitzea da. Azpi-programa hauetako bakoitzak duen izenburuaren araberako eginkizun dauka.

Lehenengo azpi-programarekin hasi aurretik 1-17 lerro bitarte horretan programa garatzeko erabiliko diren aldagaiak definitzen dira. Aldagai hauen array estatikoak

izango dira eta datu gehiagoko array bat sartzen bada balio hauek handitu beharko dira programak aurrera egiteko. Hala ere, egindako frogetan 700 puntu lortu dira gutxi gora behera, beraz 3000 puntuko array batekin ez da arazorik egongo printzipioz.

Aldagaiak definitu ostean lehenengo azpi-programa dago (19-26 lerroak), SarrerakoPuntuakHasieratu izenekoa. Izenak adierazten duen bezala, azpi-programa honen lana aurrez definitutako 3000 puntuko array-etan puntu guztien balioa hasieratzea da (-1000000.000000, -1000000.000000) koordenatuekin. Aipatzekoa da programa guztian zehar 6 hamartar erabiltzen direla, hau zenbaki errealei erreferentzia egiten zaielako da.

Bigarren azpi-programa DatuenFitxategiaIrakurri (28-51 lerroak) deitzen da eta hemen irakurtzea nahi dugun fitxategiaren izena idazteko eskatzen du, programa kokatzen den karpeta berean bilatzeko. Azpi-programa honek fitxategirik aurkitzen ez badu mezu bat bidaliko du hau esanez, bestela hasieran definitutako puntuen array-ean irakurritako puntuen koordenatuak (-1000000.000000, -1000000.000000)-ekin ordezkatuko ditu. Esan behar da, ‘ Concatenar ’ erabiliz puntuak koordenatu bezala daudela excel orrian, baina hauek .txt fitxategi moduan gorde behar dira programa honetan erabiltzeko.

Segidan dauden bi azpi-programak PuntuKopuruaZenbatu (53-58 lerroak) eta PuntuenArtekoPausuaZehaztu (60-71 lerroak) dira. Lehenengo honek aurretik aldatutako array-a irakurtzen du (-1000000.000000, -1000000.000000)-ra iritsi arte eta hori da puntu kopuruaren balioa. Honetaz baliatuz bigarrenak puntu kopuruaren arabera bi puntuen arteko distantziarik txikiena, hau da, pausua kalkulatzen du.

Programa nagusiak deitzen duen hurrengo azpi-programa SegmentuenSorrera (114-138 lerroak) deitzen delakoa da. Azpi-programa hau 118.lerrotik 125-era segmentuen egituraren hasieratzea egiten du ondoren erabiltzeko. 127-136 lerro bitartean gertatzen da puntuak lotzen dituen segmentuen sorrera. Segmentu hauek sortzeko lehenik eta behin puntu baten inguruan eduki ditzakeen zortzi puntu posibleak badauden konprobatzen da PuntuHauExistitzenDa (104-112 lerroak) azpi-programaren bitartez. Puntuaren existentziak baiezkoa ematen badu SegmentuaSortu (97-102 lerroak) azpi-programara pasatzen da bi puntuak lotzen dituen segmentua sortzeko. Segmentuak sortzeko azpi-programa honetan 0^{o}, 45^{o}, 90^{o} eta 135^{o}-ko angelua duten

segmentuak sortzen dira, eta 180º edo handiagokoei 180º kentzen zaizkie geroagoko optimizazioa errazteko. Azken azpi-programa honetan SegmentuaListanSartu (85-95 lerroak) izeneko beste azpi-programa bati deitzen zaio, honek zerrendan oraindik ez dagoen segmentua sartzen du aurretik aipatutako angeluen irizpidearekin. Segmentua zerrendan dagoen edo ez jakiteko SegmentuaZerrendanDago (73-83 lerroak) azpi-programa erabiltzen da SegmentuaListanSartu-ren barruan.

Segmentuen Sorreraren ostean SegmentuenOptimizazio (140-169 lerroak) dator, lerro berean dauden segmentuak elkartu eta ahalik eta segmenturik luzeenak sortzen dituen azpi-programa.

Azkeneko bi azpi-programak DatFitxategiarenSorrera (171-186 lerroak) eta DxfFitxategiarenSorrera (188-223 lerroak) dira, hauen eginkizun soila orain arte egindakoa .dat eta .dxf bat sortu eta bertan gordetzea da. Beraz, hau da C++ erabiliz programatutako kodearen bitartez azkenean erabiliko dena, hots, .dxf formatuko fitxategia Autocad bidez zabalduko dena [12].

Behin Autocad softwarea zabalduta bistakoa da ingeraden erpinetan soberan dauden segmentuak ezabatu behar direla. Aldi berean segmentu batzuk ezin dira ezabatu beste ingerada bat sortu dezaketelako, kasu horietan software honek bi segmentu edo gehiagoren artean soberan dagoena moztea ahalbidetzen du ‘ Recortar ’ komandoaren bidez. Azkenean artxibo hau .dwg bezala gordetzen da, aurreko kasuan bezala Solid Edge bidez zabaltzeko [11].

Solid Edge zabaldu bezain pronto Autocad bidez editatutako fitxategia zabalduko da beste kasuan bezala ‘ *iso metric part.par* ’ hautatuz ISO norma errespetatzeko. Hurrengo pausua beste kasuetan egiten den bera da, hots, saguaren eskuineko botoia sakatu eta ‘ Pasar a Ordenado ’ aukeratzea. Aurreko bideekiko desberdintasuna hemen dator, izan ere, kasu honetan ingeradak zehaztuta daude beraz zuzenean lodiera emateari ekingo zaioz. Horretarako bai erabiliko da aurrez erabilitako komandoa, hau da, ‘ Extrudir ’ eta hau aukeratzean ‘ Seleccionar Cadena ’ erabiliko da ingeradak aukeratzeko. Azken aukera hau eginda lodiera zehazten da eta zuzenean solidoa lortu. Hemendik aurrera aurreko pausu berak jarraitu behar dira, lehenengo simetria aplikatzea eta ondoren ‘ Redondear - Todos los acuerdos ’ erabiliz ingeraden irregulartasunak biribilduko dira. Biribilketen neurria 1mm-ko erradiokoa izango da,

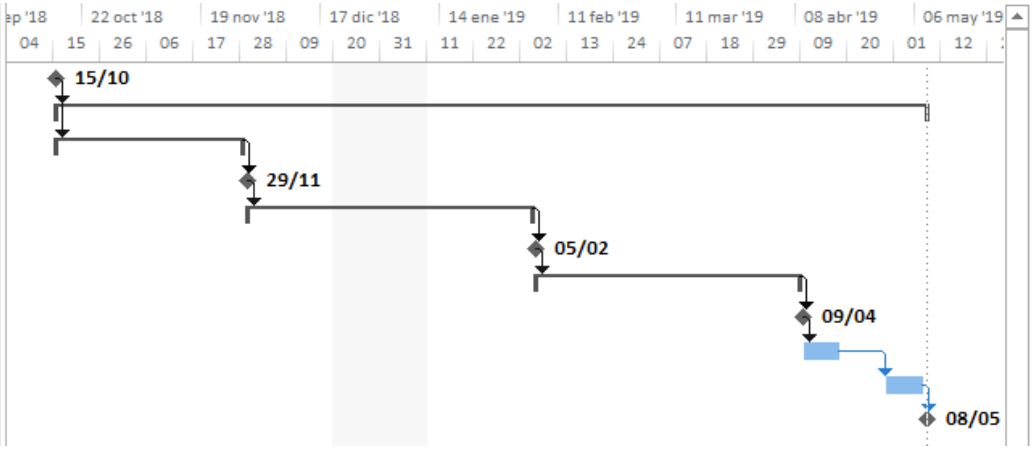
aurreko kasuko berbera, eta aplikatzearen arrazoia tentsio kontzentrazioena da baita ere.

Amaitzeko beste bi kasuetan bezala artxibo hau .stl formatuan gordeko da solidoa lortzeko. Izan ere, inprimagailuak irakurtzen duen formatua mota honetakoa da [10].

10 GANTT DIAGRAMA

Proiektu hau burutzeko jarraitu diren zeregin desberdinak adierazten dira azpiko Gantt diagraman (Irudia 5). Lanaren iraupen totala 113 egunekoa izango da, jai egun, asteburu eta azterketatako datak kontuan hartzen ez direlarik. Aipatzekoa da asteko 5 egunetan ez direla 8 orduko saioak egin. Esan bezala proiektuaren iraupena 113 egunekoa den arren, aste desberdinetan egun konkretuetan lan egin da bakoitzean 4 ordu inguru sartuz. Bai Matlab eta Excel bidezko lana, 25 egun, bai software bidezko analisia, 42 egun, ez dira asteko 5 egunetan lan eginez burutu. Izan ere, azterketa partzialak eta bestelako lanak tartean daudelarik egun eta ordu konkretu batzuetan proiektua geldirik utzi behar izan da. Guztia kontuan izanik 113 egun horietan 6 kredituak betetzeko lana burutu da gutxi gora behera.

	Zeregina	Iraupena	Hasiera data	Amaiera data
1	**Lanaren esleipena**	**1 día**	**lun 15/10/18**	lun 15/10/18
2	**Lanaren garapen praktikoa eta teorikoa**	**113 días**	mar 16/10/18	mié 08/05/19
3	***Gaiaren testuinguruaren ulermena***	**30 días**	mar 16/10/18	mié 28/11/18
6	Zuzendariarekin bilera	1 día	jue 29/11/18	jue 29/11/18
7	***Matlab eta Excel bidezko lana***	**25 días**	vie 30/11/18	lun 04/02/19
11	Zuzendariarekin bilera	1 día	mar 05/02/19	mar 05/02/19
12	***Software bidezko analisia***	**42 días**	mié 06/02/19	lun 08/04/19
16	Zuzendariarekin bilera	1 día	mar 09/04/19	mar 09/04/19
17	Emaitzen analisia	6 días	mié 10/04/19	mié 17/04/19
18	Etorkizuneko ekarpenen azterketa	6 días	lun 29/04/19	mar 07/05/19
19	**Lanaren amaiera**	**1 día**	mié 08/05/19	**mié 08/05/19**



Irudia 5. Zeregin nagusien Gantt diagrama.

Zuzendariarekin egindako bilerak lanaren mugarritzat hartu dira proiektuarekin aurrera egiteko. Ikusten den bezala zeregin nagusien Gantt diagrama zenbat ataletan banatuta dago.

Lana esleitzearekin batera proiektuaren prozesua hasten da. Lehenekin eta behin gaiaren testuingurua ulertzea proiektuaren muina da, bukatzean zuzendariarekin ikasitakoaz jardun eta aurrera jarraitzeko. Segidan, atal praktikoarekin jarraitzen da Excel eta Matlab bidezko lana burutuz. Atal honekin bukatzean zuzendariari emaitzak erakusten zaizkio lanarekin jarraitu ahal izateko. Ondoren, software bidezko analisia egiten da atal praktikoan lortutako emaitzak landuz. Lortutako azken emaitza hauek berriz ere zuzendariarekin partekatzen dira. Lana amaitutzat emateko, emaitzak eta etorkizunean eduki ditzakeen ekarpenak aztertzen dira.

10.1 GANTT DIAGRAMAREN SAKONTZEA

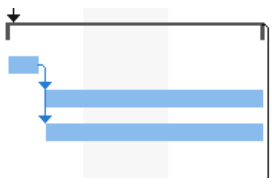
Proiektua testuinguruan kokatu eta ulertzeko bi gai nagusi eta beharrezkoak menperatu behar dira. Alde batetik optimizazio topologikoa, eta bestetik, fabrikazio gehigarria. Bi gai hauek lantzean zuzendariarekin bilera egiten da ordura arte landutakoa eztabaidatu ahal izateko (Irudia 6).

Gaiaren testuinguruaren ulermena	**30 días**	mar 16/10/18	mié 28/11/18
Optimizazio topologikoa	14 días	mar 16/10/18	mar 06/11/18
Fabrikazio gehigarria	16 días	mié 07/11/18	mié 28/11/18

Irudia 6. Gaiaren testuinguruaren zereginaren Gantt diagramaren sakontzea.

Atal praktikoari begira, hau da, Matlab eta Excel bidezko zeregina aztertuz ikusten da programazio lanak burutzen direla. Hasiera batean software hauek ezagutu behar dira. Eta hauek ezagutu ostean, esan bezala programatzen hasten da nahi edo espero diren emaitzak lortzen direla ikusten den arte. Programazio lanetan C++ softwarea ere sartzen dela aipatu behar da, besteak hasiera batean indar gehiago duten arren (Irudia 7).

Matlab eta Excel bidezko lana	**25 días**	vie 30/11/18	lun 04/02/19
Softwareak ezagutu eta lana antolatu	6 días	vie 30/11/18	vie 07/12/18
Programazio lanak	19 días	lun 10/12/18	lun 04/02/19
Optimizazio topologikoaren emaitzen analisia eta dokumentazioa	19 días	lun 10/12/18	lun 04/02/19

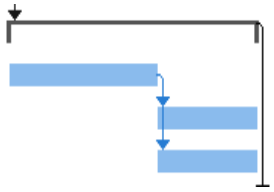


Irudia 7. Matlab eta Excel bidezko lanaren zereginaren Gantt diagramaren sakontzea.

Software bidezko analisi hau CAD artxiboa lortzeko programaren inguru koa da, baita 3D inpresioa egiteko ere, atal hau erabiliko baita. Horregatik zeregin hauetan esandako

CAD artxiboa sortu eta egituraren analisia egingo da, azkenik egitura formatu solidoan lortu arte (Irudia 8).

Software bidezko analisia	**42 días**	mié 06/02/19	lun 08/04/19
CAD softwarea ezagutu eta ulertu	27 días	mié 06/02/19	jue 14/03/19
Optimizazio egituraren analisia	15 días	vie 15/03/19	lun 08/04/19
Egituren dokumentazioa	15 días	vie 15/03/19	lun 08/04/19

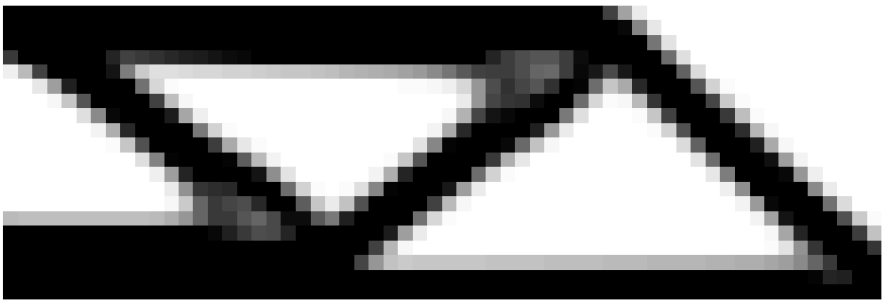


Irudia 8. Software bidezko analisiaren zereginaren Gantt diagramaren sakontzea.

Proiektuarekin amaitzeko geratzen diren zereginak emaitzen analisi eta etorkizuneko ekarpenen azterketa dira, zeregin nagusietan azaldu diren moduan. Lan guzti hau burututzat ematean zuzendariarekin partekatu eta lana amaitutzat ematen da.

11 EMAITZAK

Atal honetan proiektuan zehar lortutako emaitzak bistaratuko dira dagozkien argibide eta azalpenekin batera. Hasiera batean datu moduan erabilitako Matlab kodeak, *'A 99 line topology optimization code written in Matlab '*, sortzen duen optimizazio topologiko bidezko erantzuna erakusten da hurrengo irudian (Irudia 9).



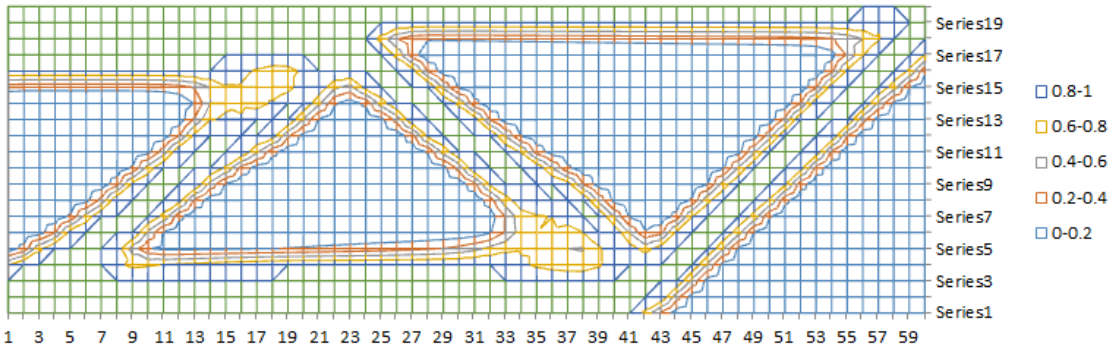
Irudia 9. Optimizazio topologiko bidez lortutako egitura erdia.

Irudi honetatik abiatuz bi bide banatzen dira. Aipatzekoa da, gaineko irudi hau x matrizearen dentsitate erlatiboak erakusten dituena dela. Aurrez azaldu den bezala beltzak 1 balioko dentsitate erlatiboa dute eta zuriak 0-koa. Esan behar da gaineko irudiko egitura edo piezaren simetrikoa falta dela ezkerreko aldean, baina kalkuluak optimizatzeko horrela egin dira guztiak.

Prozesu guztian zehar jarraitutako bide paraleloak desberdinak diren moduan, emaitzak ere kasu bakoitzak bereak dauzka eta azpiko atalen bitartez azalduko dira.

11.1 X matrizetik abiatuz lortutako emaitzak

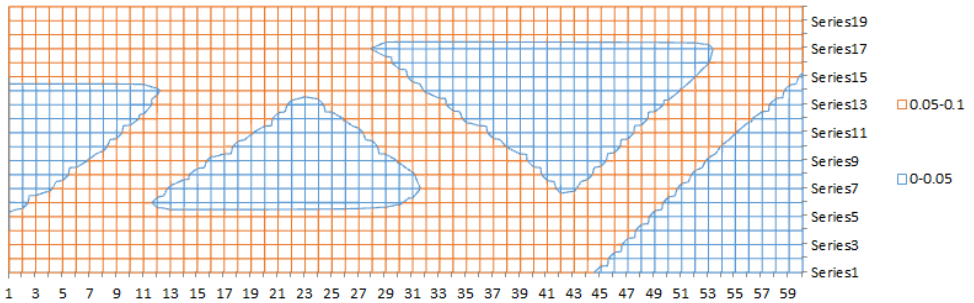
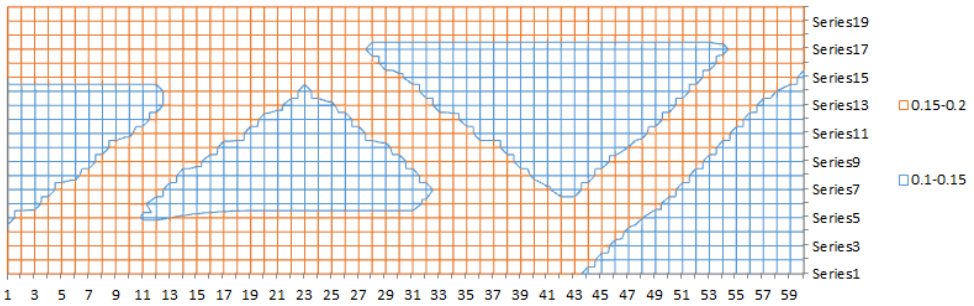
X matrizetik lortzen diren lehenengo emaitza excel bidez egindako 3 dimentsiotako grafikoaren goiko bista da, hau azpiko irudian (Irudia 10) dago ikusgai. Irudian ikusi daitekeen modura 0.2 unitateko desberdintasuna dago marra batetik bestera, honela ez da definizio eremu garbi bat ikusten. Hori dela eta, goiko bista honi aurretik aipatu den moduan barruranzko ardatzean ebaketak egin dira 0.1 unitateko desberdintasuna duten balioen arten. Hau eginik hainbat marra agertu beharrean bakar bat agertzen da inguruak argiago zehaztuz.

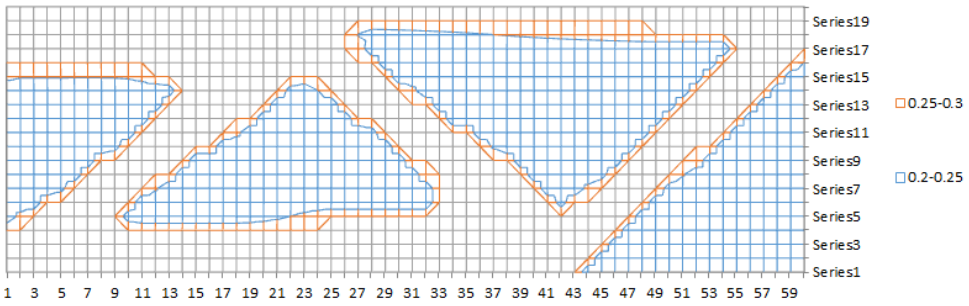
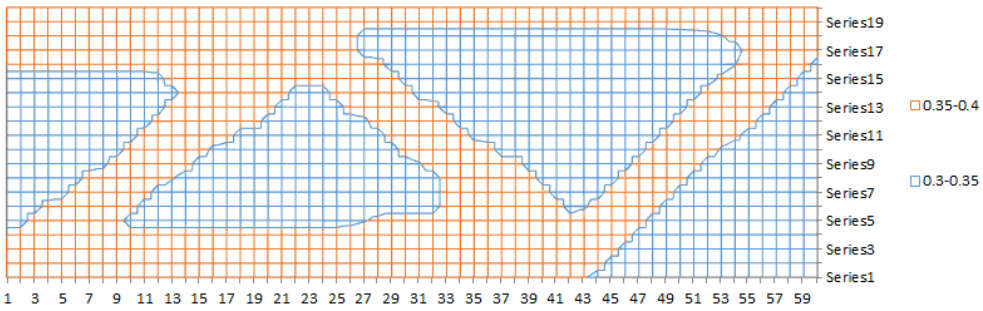
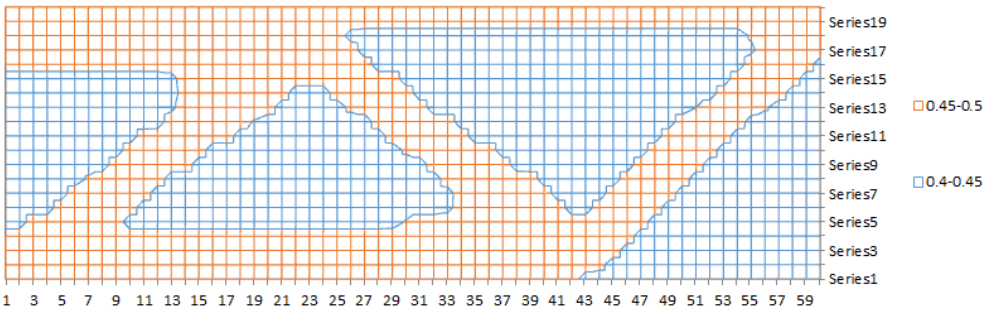
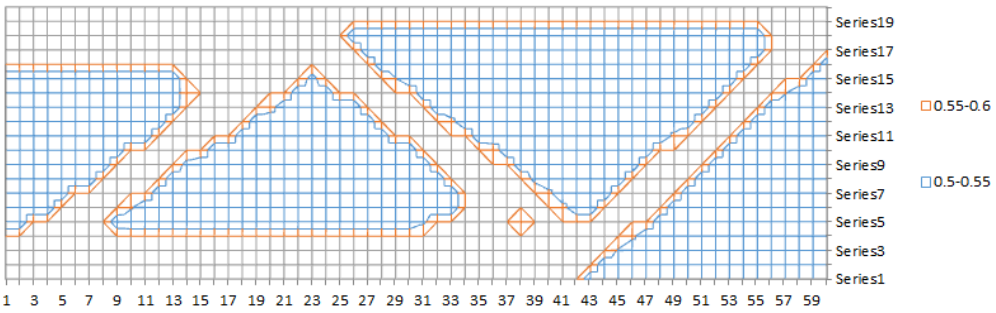
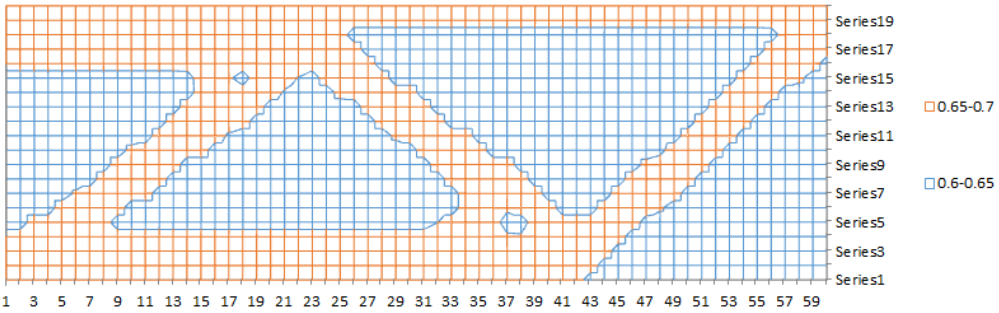


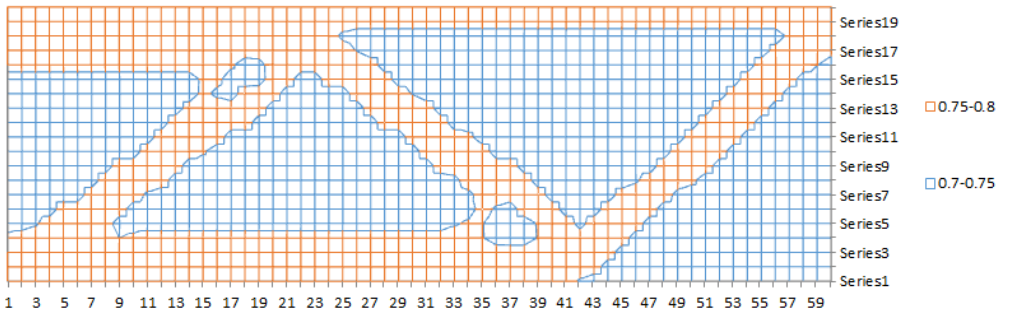
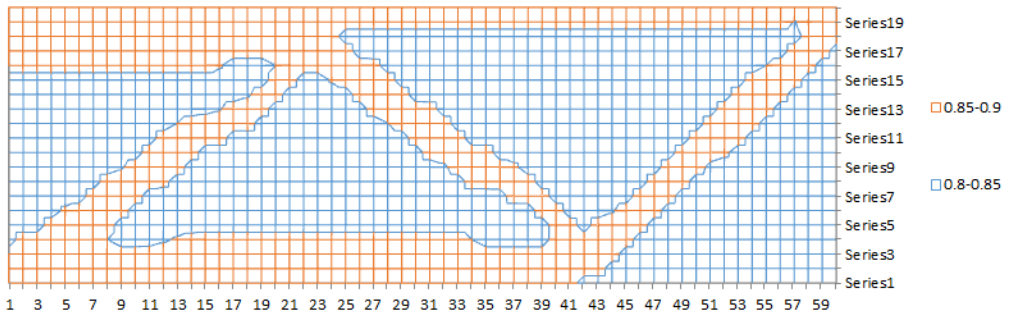
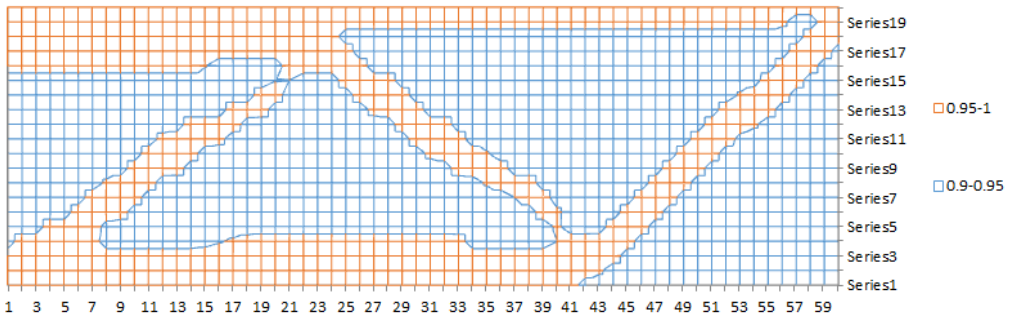
Irudia 10. X matrizearen goiko bista.

Azpiko irudi zerrendan aurretik aipatutako ebaketak agertzen dira (Taula 1). Aipatu behar da balio baxuenak kontserbakorrenak direla, eta aldi berean balio altuenak arrisku handienekoak.

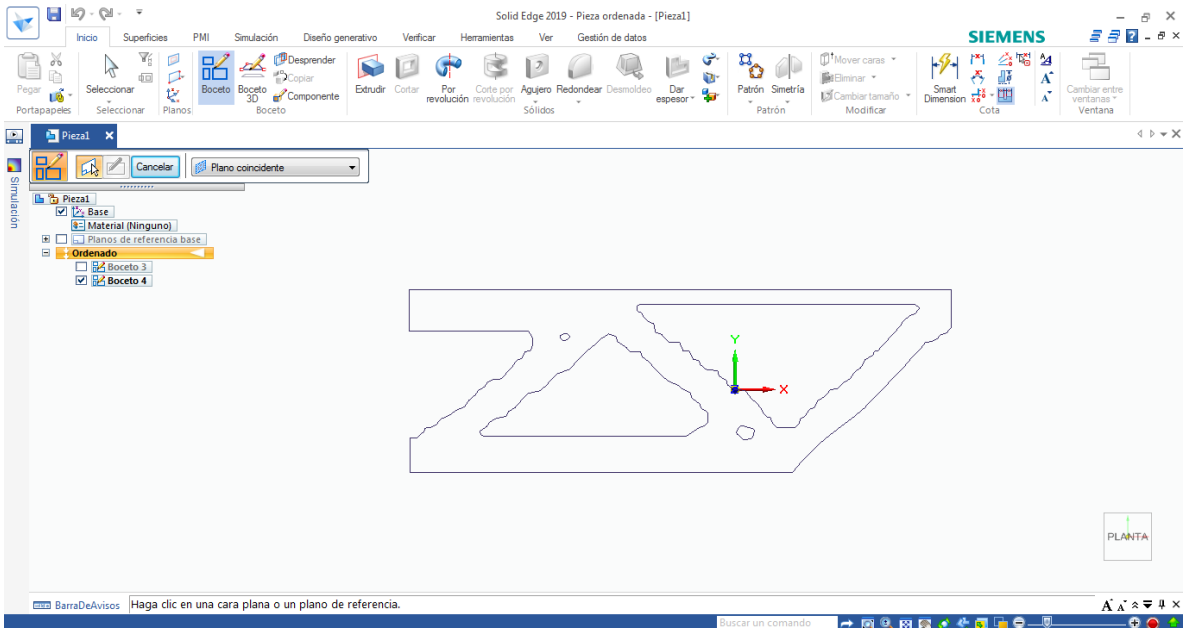
Taula 1. X matrizearen goiko bistaren ebaketak.

Balio tartea	Goiko bistaren ebaketa
0-0.1	
0.1-0.2	

0.2-0.3	
0.3-0.4	
0.4-0.5	
0.5-0.6	
0.6-0.7	

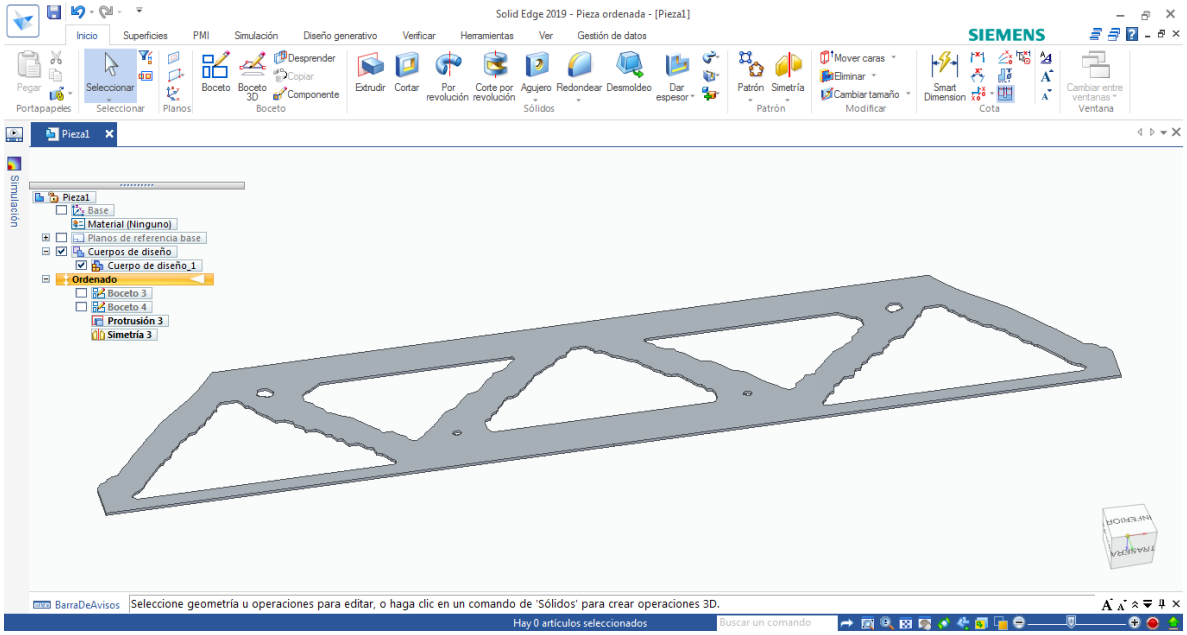
0.7-0.8	
0.8-0.9	
0.9-1.0	

Goiko bistaren ebaketa hauek ikusita, nabarmentzekoa da azkeneko irudian, 0.9-1.0 tarteko balioak aurkezten dituenean, egitura apurtu egingo dela, beraz ez da egokia balio tarte hau aukeratzea. Esan behar da, balio kontserbakorregiak hartzea ere ez dela egokiena egitura optimo bat lortu nahi den aldetik. Beraz guzti hau ikusita tarteko balioak hartzea da egokiena, honela 0.6-0.7 balio tarteko irudia aukeratu da Solid Edge bitartez marrazteko. Solid Edge-n marraztutako ingerada edo inguruen marrazkia, excel-eko bistaren gainean egindakoa, azpian ikusi daiteke (Irudia 11).



Irudia 11. 0.6-0.7 balio tarteko bistaren inguruen marrazkia Solid Edge-n.

Aurreko iruditik abiatuz, zirriborroari lodiera eman eta simetria aplikatuz bilatzen den pieza lortzen da. Kasu partikular honetan eman zaion lodiera 1 mm-koa izan da eta simetria gaineko irudiko ezkerreko plano bertikalean aplikatu da. Amaitzeko esan behar da azpiko hau dela X matrizetik abiatuz lortzen den piezaren azken itxura (Irudia 12).

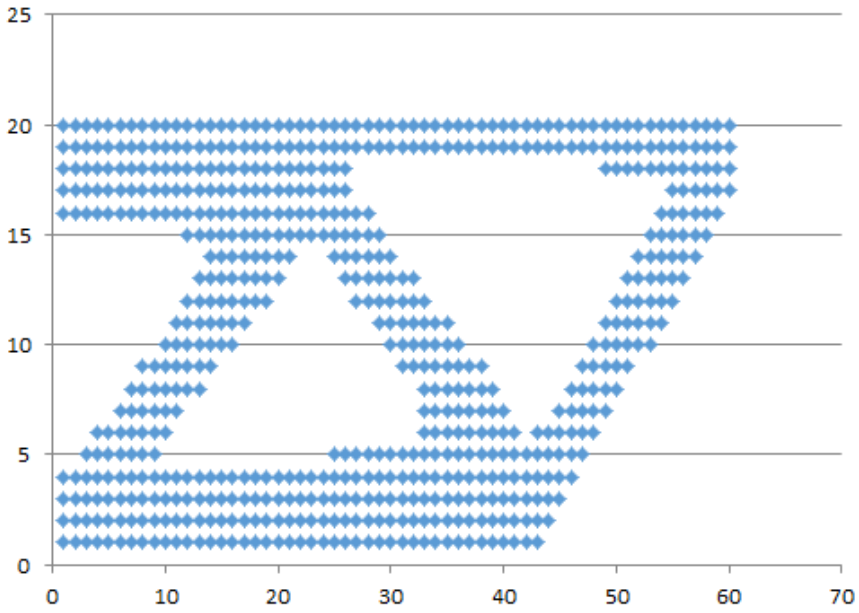
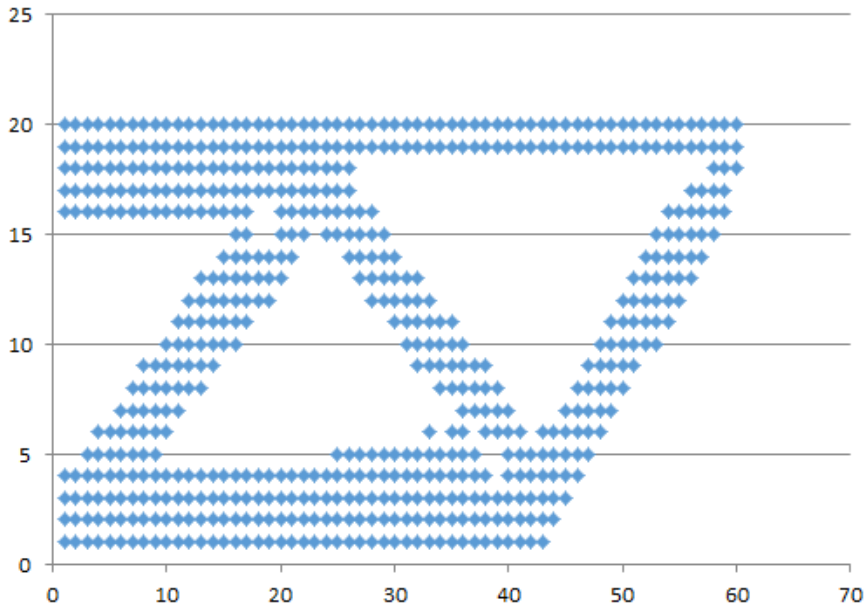


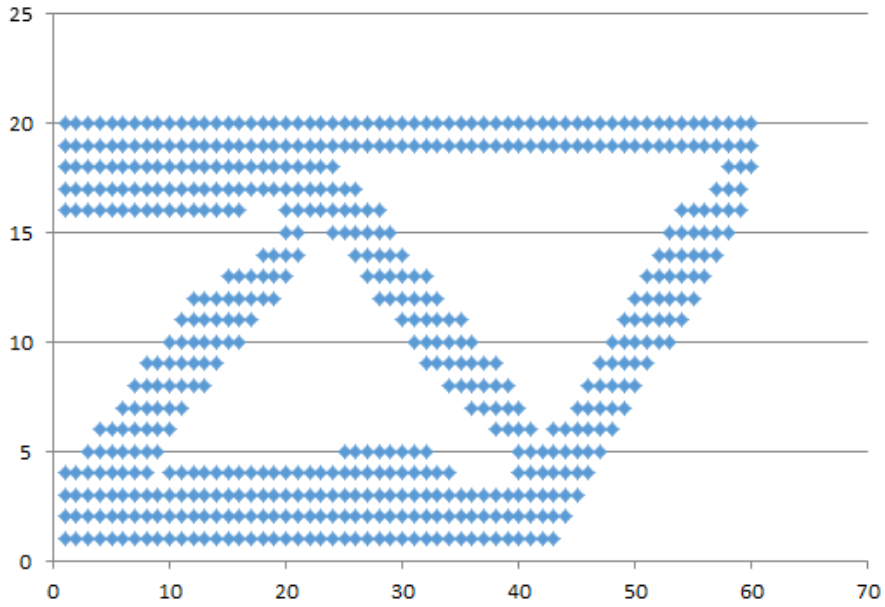
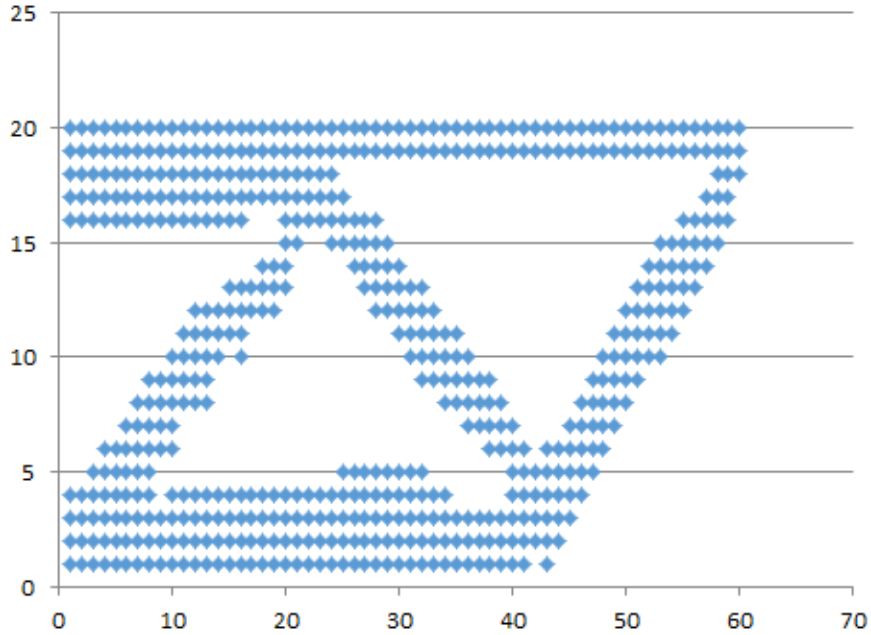
Irudia 12. Pieza osoa (X matrizetik abiatuz).

11.2 Matlab bidezko azpi-programatik abiatuz lortutako emaitzak

Matlab bidez egindako azpi-programa exekutatzean excel bidez lortzen diren grafikoak azpiko taulan ikusten dira (Taula 2). Grafiko bakoitzak dagokion ezaugarriekin desberdinduko da, azpi-programan konparaketan egitean balioak desberdinak badira edo handipenen bat duten beraien artean.

Taula 2. Azpi-programatik lortutako grafikoak.

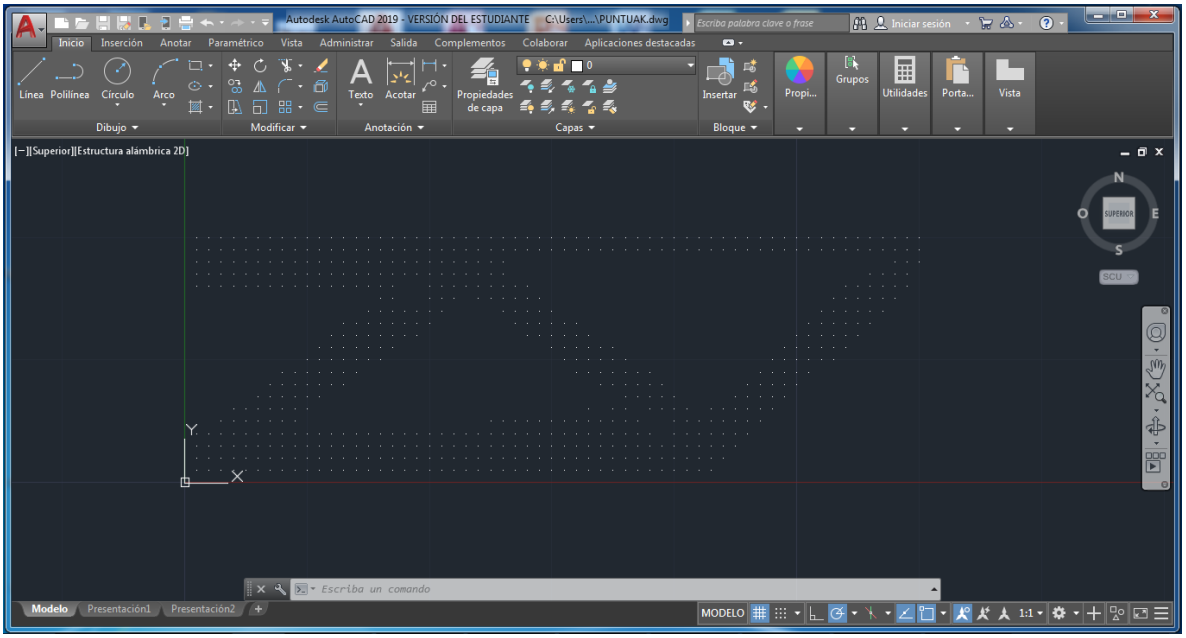
Baldintza	Grafikoa
Desberdin	
Handipena 1	

Handipena 2	
Handipena 3	

Goiko taulan ikusten diren lau kasuak aztertuz, aurretik aipatu den moduan desberdintasunaren baldintza erabiltzen den azpi-programan lortutako grafikoak pieza kontserbakorrena erakusten du. Bestalde gehien arriesgatzen den pieza 3 aldiz handiagoak diren balioak konparatzen direna izango da. Tarteko balioak kontuan hartuz bietatik handipenik gabekoa aukeratzen da. Aukera honen arrazoi nagusia, optimizazio topologiko bidez lortutako emaitzaren (Irudia 9) antzekoena dela da, gune grisetako dentsitate erlatiboak duten balioaaren arabera gehiago edo gutxiago ikusten direlarik.

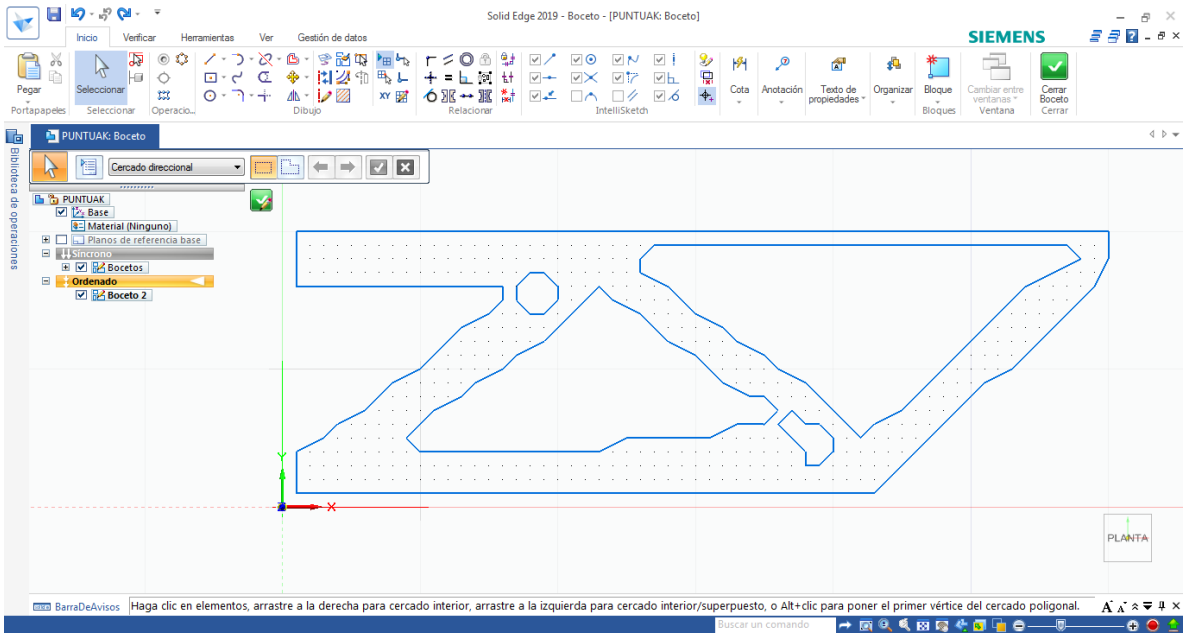
11.2.1 Puntuak zuzenean erabiliz

Prozeduraren atalean esplikatu den moduan, Autocad softwarean sartutako puntuak ikusten dira azpiko irudian (Irudia 13). Esan behar da excel bidez lortutako grafikoaren puntu berak direla, jakina den moduan.



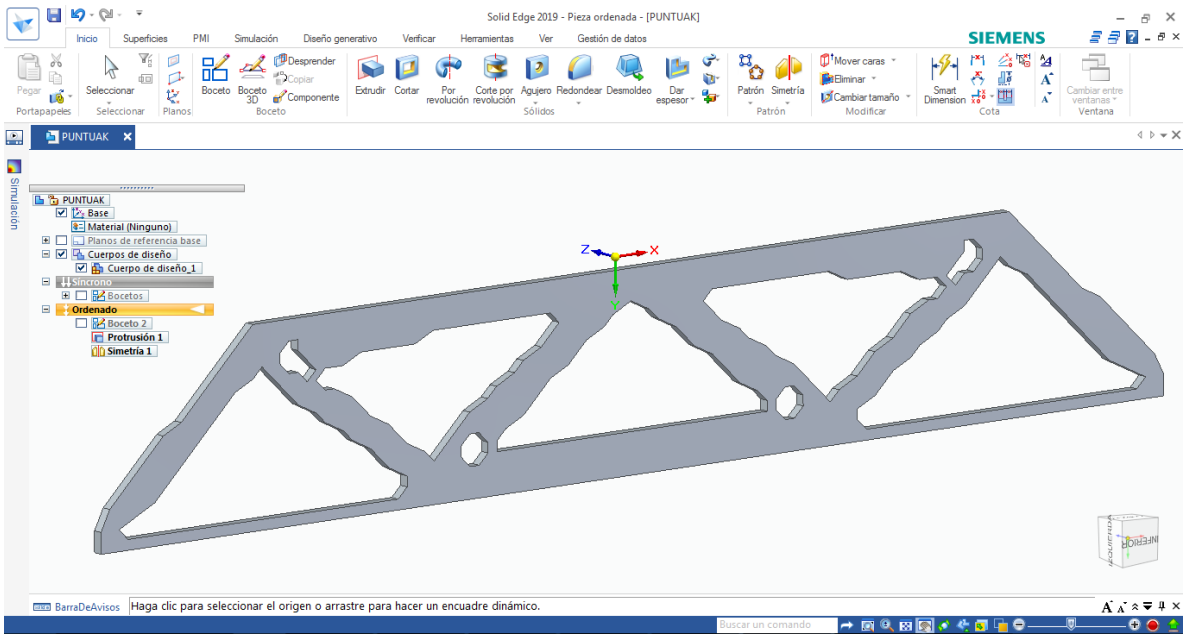
Irudia 13. Autocad bidezko handipenik gabeko puntuen irudikapena.

Azken Autocad artxiboa Solid Edge bidez zabaltzean beheko irudian (Irudia 14) ikusten diren puntuak bistaretzen dira soilik, hau da, gaineko argazkiko gauza bera ikusten da. Hala ere, azpiko irudian dagoeneko puntuak lerro zuzen eta spline kurben bitartez elkartuta daude piezaren zirriborroa osatzeko, honela piezari itxura ematen zaio. Beraz, hau da azken piezaren ingeradak erakusten dituen marrazkia.



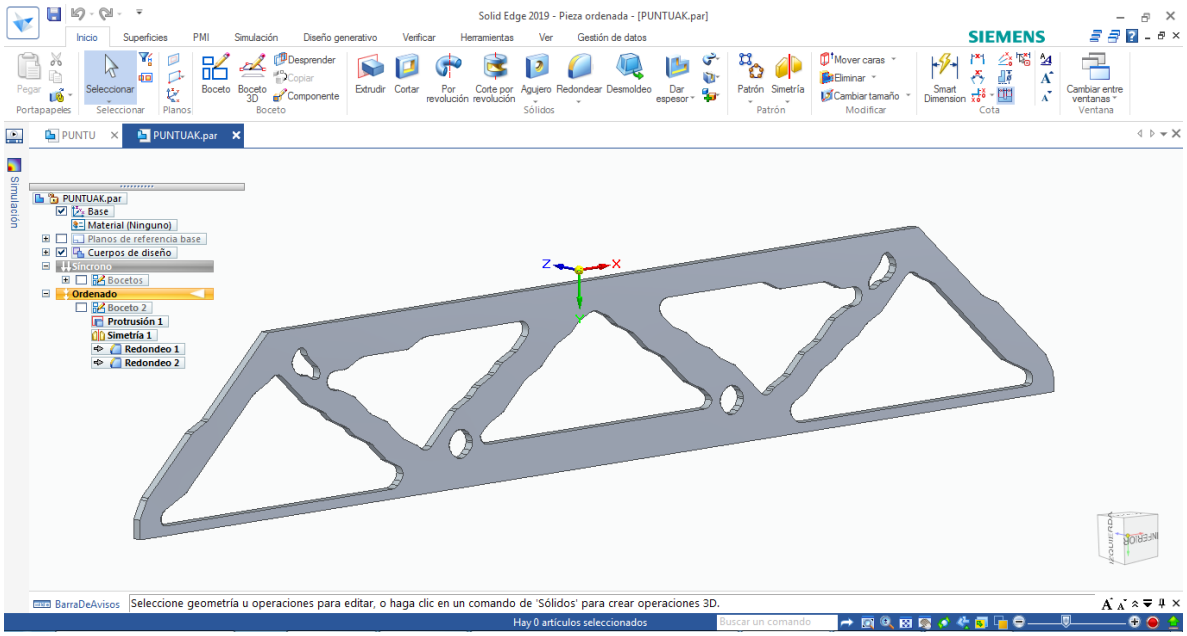
Irudia 14. Puntuen inguruko ingeraden marrazkia Solid Edge-n.

Inguruko ingeradak erakusten dituen marrazkiari lodiera eman eta simetria aplikatuz azpiko pieza osoa lortzen da, geroago inprimatuko dena. Kasu honetan ere aurrekoan bezalaxe 1 mm-ko lodiera emango zaio piezari eta simetria gaineko irudiko ezker plano bertikalarekiko aplikatuko da (Irudia 15).



Irudia 15. Pieza osoa (azpi-programatik abiatu eta puntuak zuzenean erabiliz).

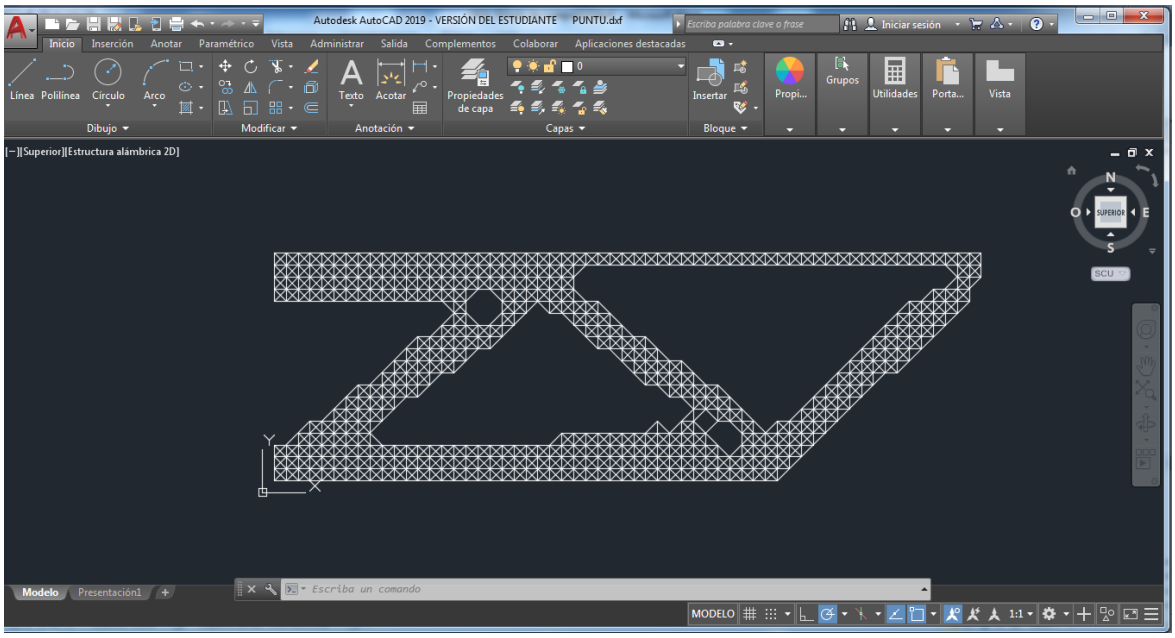
Bukatzeko esan behar da azpian ikusten den Irudia 16 dela azken bertsioa, hots, inprimagailura bidaliko dena aurreko irudiari biribilketak aplikatu ostean.



Irudia 16. Azkeneko pieza osoa (azpi-programatik abiatu eta puntuak zuzenean erabiliz).

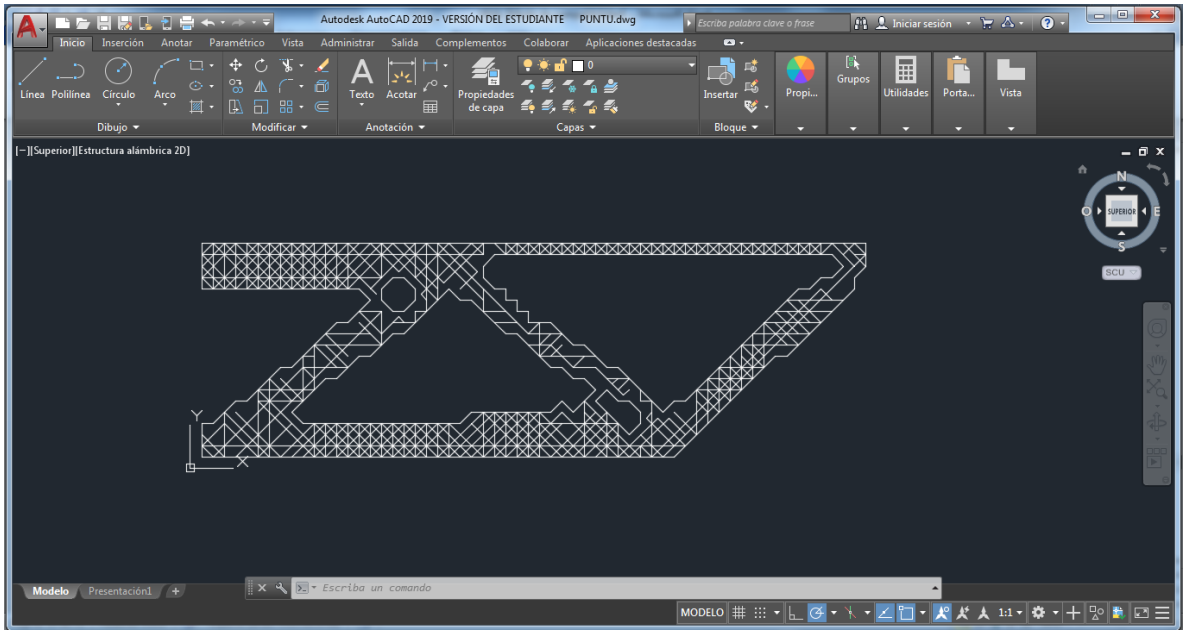
11.2.2 Puntuak segmentuen bidez lotzeko programatuz

Puntuak eduki ostean C++ bidez garatutako programa erabiliz azpiko irudiko egitura lortzen da (Irudia 17), hau da Autocad bidez ikusten dena. Aipatzekoa da segmentuak karratu baten barruan lau hiruki berdin sortzen dituztela, honela sare moduko baten itxura hartuz.



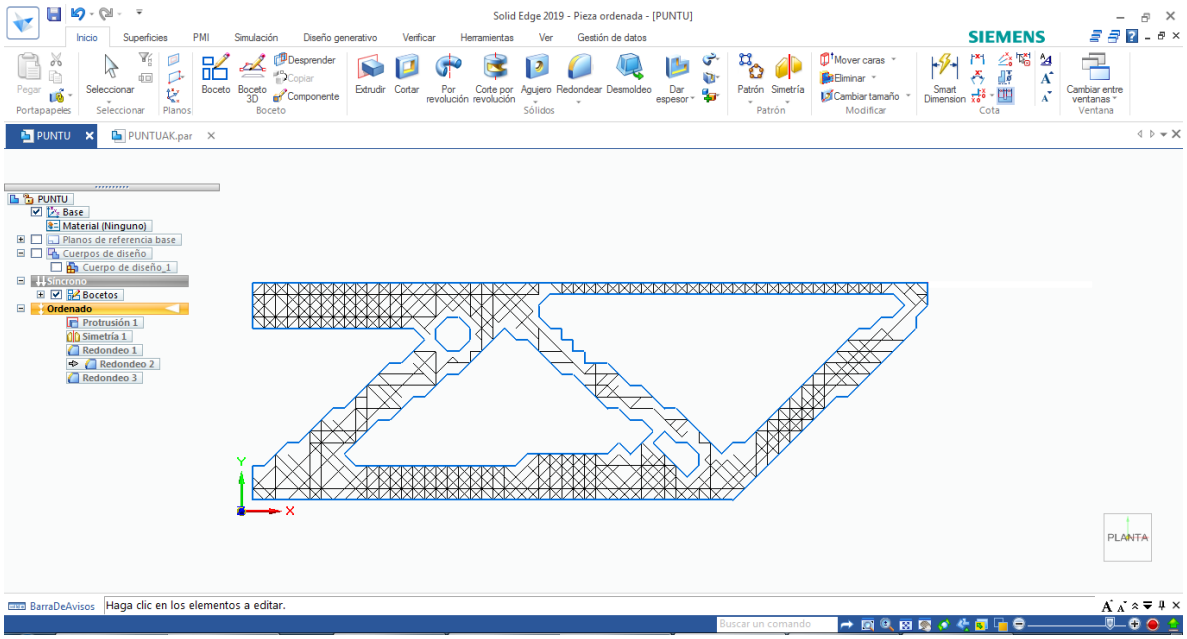
Irudia 17. C++ bidez lortutako segmentu bidezko egitura.

Ingeradak lortu nahi diren aldetik, erpinetan jarraitu nahi diren bideko segmentuak utziko dira soilik, eta prozeduran azaldu den moduan ebakitzen diren segmentuetan soberan dagoen zatia kenduko da. Azken bi ekintza hauek gauzatzean Irudia 18 honetan ikusten dena lortzen da.



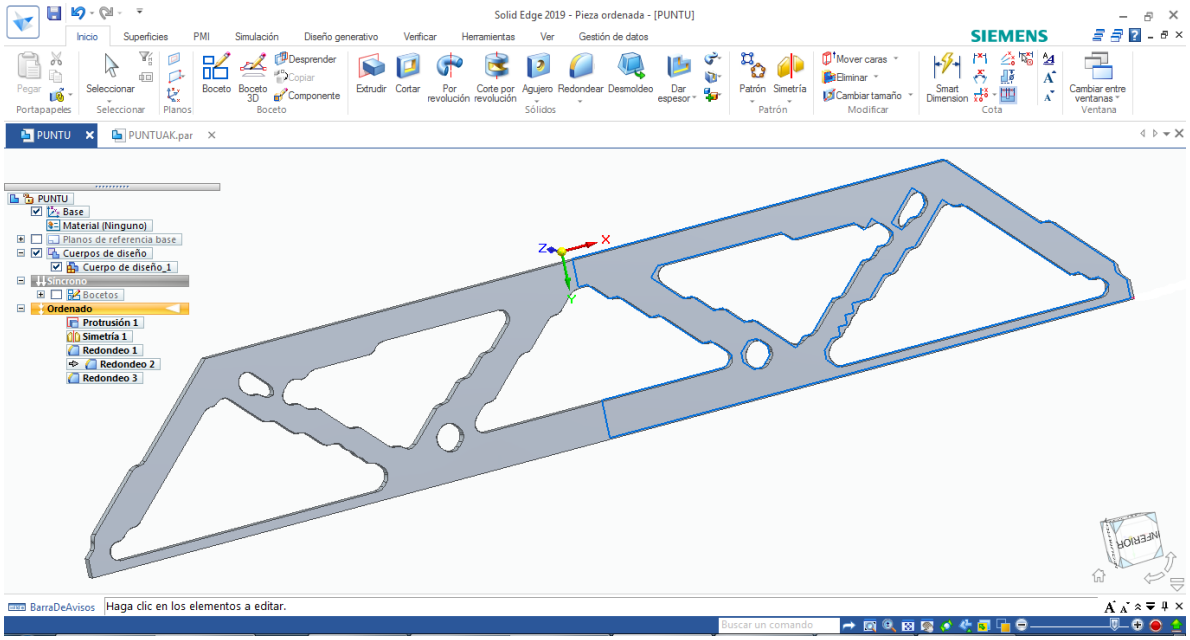
Irudia 18. Behar ez diren segmentuak kentzean lortutako egitura.

Goikaldean iksuten den irudia Solid Edge bitartez zabaltzean ikusten da inguru edo ingeradak nola igartzen dituen. Hau gainean klitatuta besterik ez da egiten, honela kate itxiak direla ziurtatzeko gainontzean lodiera ezingo zaiolako eman.



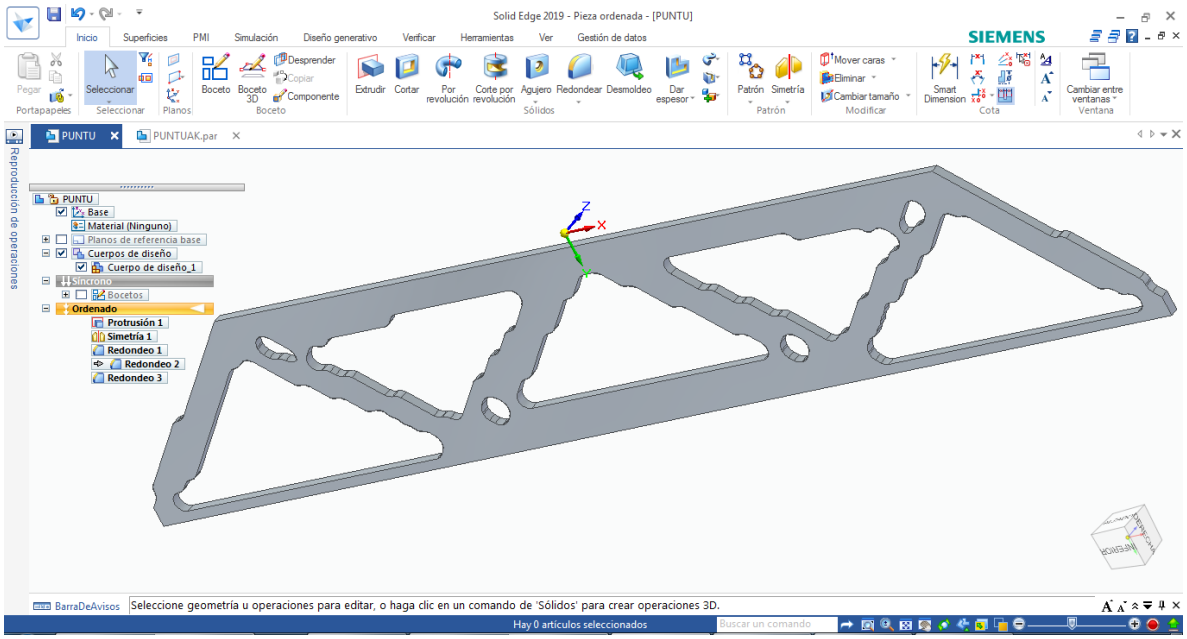
Irudia 19. Ingeraden aukeraketan Solid Edge-n.

Gaineko irudiari 1mm-ko lodiera eman eta simetria aplikatzean azpiko pieza edo egitura (Irudia 20) lortzen da. Aipatu beharrekoa da, azpiko irudian urdinez ikusten den ingerada biribilketak jasan aurretik dagoena dela.



Irudia 20. Pieza osoa urdinez biribilketarik gabekoa ageri dela.

Prozeduran zehar esan bezala 1mm-ko erradioko biribilketan egiten dira pieza osoaren ertz guztietan. Honela azpian dagoen azken pieza lortzen da (Irudia 21) eta ikusten den moduan ingeradak askoz ere suabeagoak bihurtzen dira. Gainera aurretik azaldu den moduan tentsio kontzentrazioak murrizteak onurak besterik ez dakartza. Amaitzeko esan behar da azpiko azken irudiko pieza izango dela inprimatzen dela.



Irudia 21. Azkeneko pieza osoa (azpi-programatik abiatu eta C++ bidez programatuz).

12 BIDE GUZTIEN ARTEKO KONPARAKETA

Lanaren atal honetan proiektuaren emaitza lortzeko erabilitako bide paraleloen konparaketa egiten da. Hiru kasu hauek beraien berdintasunak dituzten arren desberdintasun nagusia hasieran antzematen da, batek zuzenean excel bidez grafikoak erakusten dituelako eta beste biak ordea Matlab bidezko azpi-programa bat erabiltzen dute excel orrialde batean grafikoa egiten den arte. Gainera azken bi bide hauek puntuak lortu ostean ere oso bide desberdinak jarraitzen dituzte.

Lehenengo bi kasuen antzekotasun nabarmena azkeneko pausuko marrazkia eskuz egiten dela da, baina bigarren kasuan Solid Edge softwareak puntuak detektatzen dituenez hauek lotzen ditu eta piezaren forma ezartzea errazagoa da. Lehenengo kasuko marrazkian ordea aipatu bezala matrizearen goiko bistaren argazkia fondoan jartzen da eta honen gainean egiten da marrazkia, beraz erroreak egiteko tartea handiagoa da, izan ere, eskuz determinatzen da zein izango den materiala duen eremua. Hirugarren kasuak ordea marrazkiari dagokionez eduki dezakeen antzekotasun bakarra segmentuen ezabaketa eskuz egiten dela, besterik ezin da esan.

Desberdintasunei dagokienez, hasteko esan behar da bi eta hirugarren kasuetan exekutatzen den azpi-programak erakusten digula materiala non dagoen. Hori dela eta, ez dago egilearengan erabaki zehatzik hartu beharrik materialaren banaketari dagokionez. Bestalde, lehenengo ibilbidean, x matrizetik abiatutakoan, excel bitartez egindako goiko bistaren ebaketen arabera erabakitzen du egileak zein materia banaketa aukeratzen duen. Gainera, x matrize bakoitzerako banaketa egokienaren konprobaketak egin behar dira. Egia da azpi-programari baldintza desberdinak jarriz materia banaketa aldatzen dela, baina hau automatikoki egiten den zerbait da eta behin baldintza egokiena aukeratuta ez dago berriz ere konprobatu beharrik.

Atal honekin bukatzeko kasu guztiak egokiak izan daitezkeela esan behar da, beti ere egoeraren arabera kasu bat edo beste aukeratuz. Hala eta guztiz ere, proiektu honetan zehar prozesua ahalik eta automatikoen egitea bilatzen denez hirugarren kasua izango da egokiena. Izan ere, materiaren dentsitate erlatiboak ezabatu eta 0-1 balioak soilik lortzea aurrerapena izan daiteke, gune grisak ez baitira oso interesgarriak. Azken gune hauek ez jakintasun eta erabaki gehigarriak hartu behar izatea besterik ez duelako

eragiten. Gainera puntuak elkarren artean automatikoki lotzea aurrerapen handia da geroago soberan daudenak ezabatu behar diren arren.

Guzti honen gainetik azken pausuan, aipatu bezala, behar ez diren segmentuen ezabaketa hori eskuz egitea ez da guztiz egokia, baina erabilitako softwareak eskainitakoaren arabera pieza lortzeko modurik azkar eta egokiena hau izango litzateke.

13 ONDORIOAK

Proiektu honetan zehar Matlab bidez garatutako optimizazio topologikoaren programa hartu eta landu da 3 dimentsiotako pieza bat inprimatzera iritsi arte. Matlab kode honek optimizazio topologikoa gauzatzen du, aurrez azaldu diren parametroen arabera, eta dentsitate erlatiboak erakusten dituen matrize bat lortzen du emaitza moduan non dentsitate erlatiboak materiala dagoen edo ez esaten duten.

Programa honetatik abiatuz bi bide banatu dira, batek excel-era jo da zuzenean, eta bestea ordea Matlab softwarean geratu da azpi-programa bat sortuz. Azpi-programa honen ondoren beste bi bidetan banatu da, bat garatutako puntuak landuz, eta bigarrena C++ bidezko programazioan sartuz. Baina azken batean bide guztiak Solid Edge-n amaitu dira 3 dimentsioetako piezak erakutsiz, emaitzen atalean ikusi den bezala.

Lortutako emaitzen harira, nahiko emaitza egokiak dira. Izan ere, Matlab bidez lortutako optimizazio topologiko bidezko lehenengo emaitzaren itxura ikusten da piezetan bertan. Horrek esan nahi du jarraitutako prozesua aproposa izan dela.

Proiektu hau gauzatzean ateratako ondorio nagusia da, pieza baten optimizazio topologikotik abiatuz modu erraz eta nahiko azkar batean pieza optimoa eskuartean edukitzea lortu daitekeela. Gainera, erabili beharreko bideak sinpleak eta edonoren eskura daudenak dira.

Azpi ondorioak ere badaude, esaterako optimizazio topologikoaren erabilgarritasunari begira. Optimizazio topologikoaren metodoari arreta jarriz, esan behar da oso erreminta baliagarria dela egitura elementuen diseinuari dagokionez. Honek fabrikazioari dagokionez material aurreztea eragiten du diseinatutako elementuaren pisua murriztuz aldi berean, baina beti ere propietate mekanikoak mantenduz.

Gaur egun existitzen diren erreminta konputazionalengatik aurrerapen asko gauzatu dira. Horietako bat Elementu Finituen Metodoari dagozkion algoritmoen azkartasuna da, proiektuaren kasuan optimizazio topologikoaren Matlab kodearen barruan kokatzen den azpi atal bat izanik. Lan hau aurrerapenekin lotzen den puntu garrantzitsuena fabrikaziorako garatutako teknologia berriak dira. Kasu honetan 3D

inprimatze bidezko fabrikazioa da metodoa eta jakina den moduan ia-ia edozein itsura eta tamainatako produktuak sortzea ahalbidetzen du.

Bukaera emateko azken aipamena, optimizazio topologikoko eta 3 dimentsiotan inprimatzeko metodoak nahiko teknika berriak direla esan behar da. Gainera etengabeko garapenean daude eta egunez egun berrikuntza nabarmenak izaten dituzte. Honek esan nahi du, etorkizun hurbil batean aurrerapen izugarriak eskainiko dituzten arloak direla.

14 ETORKIZUNEKO EKARPENAK

Lan hau kanpo indar konstante puntual bat jasaten duten egituren optimizazio topologikoan oinarritu da soilik. Egiturak jasaten duen karga bakarra kanpo indar hau kontsideratuz, zenbait balio ez dira kontuan izan, adibidez karga horrek eragin desberdina duela modu jarrai edo ez jarraian aplikatzen denean. Lanean karga konstantetzat hartu da esan bezala, baina kasu errealean hau ez da horrela izango. Beraz etorkizunean honen inguruko hausnarketa egin eta optimizazio topologikoa hobetzea aukera bat izango litzateke.

Esan bezala, etorkizuneko proiektu modura aztertu diren karga puntualez bestelakoak egotearen aukera ikastea ere proposatzen da. Bestelako karga hauek adibidez termikoak izan daitezke edo besterik gabe puntualak izan beharrean sakabanatuak. Bestalde karga hauek aplikatzean kontuan izan behar dira erabilitako materiala, izan ere, eragin handia dauka.

Etorkizunean optimizazio topologikoan egin behar denaren ingurukoa alde batera utzita, ingeraden interpretazioaren inguruko hobekuntzak aipatuko dira. Hasteko ingeradak lortzeko erari dagokionez oraindik eta modu automatikoagoa lortzea posible izango litzateke. Automatikotasun hau C++ bidez garatutakoarekin zuzenean inguruak igarriko balira, Solid Edge bitartez, dagoeneko pieza egiteko prest egongo litzateke. Horretarako aukera bat da Matlab bidezko azpi-programak sortutako puntuak zenbat segmentu batzen dituzten kalkulatzea. Honen bitartez puntuan bertan 3, 4, 6 edo 7 segmentu batzen badira soberan daudenak ezabatu beharko lirateke, soilik ingurua sortzen duten segmentuak utziz.

Ingeradak hobetzeko beste aukera bat puntuak lotzeko erabiltzen diren elementuak segmentu zuzenak izan partez, hauek ingurua interpretatu eta kurbatura leuntzen badute aurrerapen nabaria izango litzateke. Proiektu honetan lortzen den biribilketa edo leunketa bakarra piezan Solid Edge bitartez aldaketak gauzatzean denez, aurretiko programazioan kurba leunak lortzea bikaina izango litzateke.

Bukatzeko esan behar da hobekuntza guzti hauek etorkizun hurbil batean beteko direnak izango direla. Proiektu guztian zehar aipatu den moduan, etengabeko

garapenean dauden alor nagusienetakoak baitira landutako optimizazio topologiko eta 3 dimentsioko inprimaketa.

BIBLIOGRAFIA

[1] Rankia. Optimización vs Eficiencia. [Online] 2016.eko. https://www.rankia.com/blog/gestion-cartera/3167733-optimizacion-vs-eficiencia.

[2] M.P. Bendsøe, O. Sigmund. *Topology Optimization. Theory,Methods, and Applications.* le : Springer-Verlag Berlin Heidelberg, 2004.

[3] S. Joshi, J.C. Medina, F. Menhorn, S. Reiz, B. Rüth, E. Wannerberg, A. Yurova. *CAD-integrated Topology Optimizational Engineering.* le : Bavarian Graduate School of Computational, 2016.

[4] *A sequential element rejection and admission (SERA) topology optimization code written in Matlab.* R. Ansola Loyola, O. M. Querin, A. Garaigordobil Jiménez, C. Alonso Gordoa. 3, 2018., Structural and Multidisciplinary Optimization, 58. bol., or. 1297-1310.

[5] O. M. Querin, M. Victoria, C. Alonso, R. Ansola, P. Martí,. *Topology Design Methods for Structural Optimization.* [ed.] Academic Press. 1. 2017. or. 204.

[6] *A 99 line topology optimization code written in Matlab.* O. Sigmund. 2, le : Structural and Multidisciplinary Optimization, 2001., 21. bol.

[7] Centro Avanzado de Tecnologias Aeroespaciales. CATEC. [Online] 2011.eko. http://www.catec.aero/es.

[8] *A case study on topology optimized design for additive manufacturing.* A. W. Gebisa, H. G. Lemu. le : IOP Publishing Ltd, 2017. IOP Conference Series: Materials Science and Engineering. 276. bol.

[9] MathWork. *MATLAB The Language of Technical Computing.* 2005.

[10] Siemens AG. *Solid Edge fundamentals.* 2011.

[11] AUTODESK Design Academy. *Manual de AutoCAD.* 2010.

[12] L. I. Olivares Flores. *Manual de Programación en Lenguaje C++.* 2008.

I. ERANSKINA - MATLAB BIDEZKO OPTIMIZAZIO TOPOLOGIKOAREN KODEA

```
%%%% A 99 LINE TOPOLOGY OPTIMIZATION CODE BY OLE SIGMUND, OCTOBER 1999 %%%
function x=top(nelx,nely,volfrac,penal,rmin)
% INITIALIZE
x(1:nely,1:nelx) = volfrac;
loop = 0;
change = 1.;
% START ITERATION
while change > 0.01
loop = loop + 1;
 xold = x;
% FE-ANALYSIS
[U]=FE(nelx,nely,x,penal);
% OBJECTIVE FUNCTION AND SENSITIVITY ANALYSIS
[KE] = lk;
c = 0.;
for ely = 1:nely
for elx = 1:nelx
n1 = (nely+1)*(elx-1)+ely;
n2 = (nely+1)* elx +ely;
Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1;2*n2+2; 2*n1+1;2*n1+2],1);
c = c + x(ely,elx)^penal*Ue'*KE*Ue;
dc(ely,elx) = -penal*x(ely,elx)^(penal-1)*Ue'*KE*Ue;
end
end
% FILTERING OF SENSITIVITIES
[dc] = check(nelx,nely,rmin,x,dc);
% DESIGN UPDATE BY THE OPTIMALITY CRITERIA METHOD
[x] = OC(nelx,nely,x,volfrac,dc);
% PRINT RESULTS
change = max(max(abs(x-xold))); disp([' It.: ' sprintf('%4i',loop) ' Obj.: '
              sprintf('%10.4f',c)
              ' Vol.: ' sprintf('%6.3f',sum(sum(x))/(nelx*nely))
              ' ch.: ' sprintf('%6.3f',change )])
% PLOT DENSITIES
colormap(gray); imagesc(-x); axis equal; axis tight; axis off;pause(1e-6);
end
%%%%%%%%%% OPTIMALITY CRITERIA UPDATE %%%%%%%%%%
function [xnew]=OC(nelx,nely,x,volfrac,dc)
l1 = 0; l2 = 100000; move = 0.2;
while (l2-l1 > 1e-4)
lmid = 0.5*(l2+l1);
xnew = max(0.001,max(x-move,min(1.,min(x+move,x.*sqrt(-dc./lmid)))));
if sum(sum(xnew)) - volfrac*nelx*nely > 0;
l1 = lmid;
else
l2 = lmid;
end
end
%%%%%%%%%% MESH-INDEPENDENCY FILTER %%%%%%%%%%%%
function [dcn]=check(nelx,nely,rmin,x,dc)
dcn=zeros(nely,nelx);
for i = 1:nelx
for j = 1:nely
sum=0.0;
```

```
for k = max(i-round(rmin),1):min(i+round(rmin),nelx)
for l = max(j-round(rmin),1):min(j+round(rmin), nely)
fac = rmin-sqrt((i-k)^2+(j-l)^2);
sum = sum+max(0,fac);
dcn(j,i) = dcn(j,i) + max(0,fac)*x(l,k)*dc(l,k);
end
end
dcn(j,i) = dcn(j,i)/(x(j,i)*sum);
end
end
%%%%%%%%%% FE-ANALYSIS %%%%%%%%%%%%
function [U]=FE(nelx,nely,x,penal)
[KE] = lk;
K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
F = sparse(2*(nely+1)*(nelx+1),1); U =sparse(2*(nely+1)*(nelx+1),1);
for ely = 1:nely
for elx = 1:nelx
n1 = (nely+1)*(elx-1)+ely;
n2 = (nely+1)* elx +ely;
edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1;2*n2+2;2*n1+1; 2*n1+2];
K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
end
end
% DEFINE LOADSAND SUPPORTS(HALF MBB-BEAM)
F(2,1) = -1;
fixeddofs = union([1:2:2*(nely+1)],[2*(nelx+1)*(nely+1)]);
alldofs = [1:2*(nely+1)*(nelx+1)];
freedofs = setdiff(alldofs,fixeddofs);
% SOLVING
U(freedofs,:) = K(freedofs,freedofs) \F(freedofs,:);
U(fixeddofs,:)= 0;
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%%
function [KE]=lk
E = 1.;
nu = 0.3;
k=[ 1/2-nu/6 1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8
-1/4+nu/12 -1/8-nu/8 nu/6 1/8-3*nu/8];
KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
```

II. ERANSKINA - PUNTUAK SEGMENTU BIDEZ LOTZEKO KODEA

```
#include <stdio.h>
#include <windows.h>

// 3000 puntu baino gehiago sartzen badira, balio hau 3 lekuetan aldatu behar da
int   gNumEltEnt = 3000;
float gValX[3000], gValY[3000];

int   gNumPts=0;
float gPaso=1000000.000000;

char gNomFch [80] = "";


// 6000 segmentu baino gehiago sortzen badira, balio hau 6 lekuetan aldatu behar da
int   gNumMaxSeg = 6000;
float gSegXIni [6000], gSegYIni [6000], gSegXFin [6000], gSegYFin [6000], gSegAng [6000];
int   gCntSeg=0;

void SarrerakoPuntuakHasieratu ()
{
    // puntuek -10000000.000000,-1000000.000000 -ra hasieratzen dira
    for (int i=0; i<gNumEltEnt; i++) {
        gValX[i]=-1000000.000000;
        gValY[i]=-1000000.000000;
    }
}

```

```
28  void DatuenFitxategiaIrakurri ()
29  {
30      FILE* fichero;
31      char nomFchDat [80] = "";
32      int i = 0;
33
34      printf("Fitxategiaren izena idatzi (luzapenik gabe) -> ");
35      scanf("%s", &gNomFch);
36      strcat(nomFchDat, gNomFch);
37      strcat(nomFchDat, ".txt");
38
39      fichero = fopen(nomFchDat, "r");
40      if (fichero==NULL) printf("Fitxategia %s ez da aurkitu\n", nomFchDat);
41      else {
42          char titulo [80];
43          fscanf(fichero,"  %s", &titulo);
44          printf("Fitxategia irakurri \"%s\"\n", nomFchDat);
45          while (! feof(fichero)){
46            fscanf(fichero,"  %f,%f", &gValX[i], &gValY[i]);
47            i++;
48          }
49          fclose(fichero);
50      }
51  }
52
```

```
53  void PuntuKopuruaZenbatu()
54  {
55      gNumPts =0;
56      while ((gNumPts<gNumEltEnt) && (gValX[gNumPts] != -1000000.000000) && (gValY[gNumPts] != -1000000.000000)) gNumPts++;
57      printf("Puntu kopurua = %d\n", gNumPts);
58  }
59
60  void PuntuenArtekoPausuaZehaztu ()
61  {
62      float inc = 0.0;
63      gPaso = 1000000.000000;
64      for (int i=1; i<gNumPts; i++) {
65          inc = abs(gValX[0] - gValX[i]);
66          if ((inc>0.0) && (inc < gPaso)) gPaso = inc;
67          inc = abs(gValY[0] - gValY[i]);
68          if ((inc>0.0) && (inc < gPaso)) gPaso = inc;
69      }
70      printf("Puntuen arteko pausua = %f\n", gPaso);
71  }
72
73  bool SegmentuaZerrendanDago (float segXIni, float segYIni, float segAng)
74  {
75      bool valor = false;
76      int  i=0;
77
```

```
78      while ((i<gNumMaxSeg) && (!valor)) {
79          if ((gSegXIni[i] == segXIni) && (gSegYIni[i] == segYIni) && (gSegAng[i] == segAng)) valor = true;
80          i++;
81      }
82      return valor;
83  }
84
85  void SegmentuaListanSartu (float segXIni, float segYIni, float segXFin, float segYFin, float segAng)
86  {
87      if (!SegmentuaZerrendanDago (segXIni, segYIni, segAng)) {
88          gSegXIni[gCntSeg] = segXIni;
89          gSegYIni[gCntSeg] = segYIni;
90          gSegXFin[gCntSeg] = segXFin;
91          gSegYFin[gCntSeg] = segYFin;
92          gSegAng [gCntSeg] = segAng;
93          gCntSeg++;
94      }
95  }
96
97  void SegmentuaSortu (float XIni, float YIni, float XFin, float YFin, float angulo)
98  {
99      //segmentu guztiak angelu hauetan sortu 0, 45, 90 0 135º
100     if ((angulo == 0.000000) || (angulo == 45.000000) || (angulo == 90.000000) || (angulo == 135.000000)) SegmentuaListanSartu (XIni, YIni, XFin, YFin, angulo);
101     else                                                                                                   SegmentuaListanSartu (XFin, YFin, XIni, YIni, angulo-180.000000);
102 }
103
104 bool PuntuHauExistitzenDa (float newX, float newY)
```

```
105 {
106     bool  valor = false;
107 
108     for (int i=0; (i<gNumPts && !valor); i++) {
109        if ((newX == gValX[i]) && (newY == gValY[i])) valor = true;
110     }
111     return valor;
112 }
113 
114 void SegmentuenSorrera ()
115 {
116     printf("Segmentuak sortu\n");
117 
118     //Segmentuen egitura hasi
119     for (int i=0; i<gNumMaxSeg; i++) {
120        gSegXIni [i] = -1000000.000000;
121        gSegYIni [i] = -1000000.000000;
122        gSegXFin [i] = -1000000.000000;
123        gSegYFin [i] = -1000000.000000;
124        gSegAng  [i] = -1000000.000000;
125     }
126     gCntSeg = 0;
127     for (int i=0; i<gNumPts; i++) {
128        if (PuntuHauExistitzenDa (gValX[i]+gPaso, gValY[i]))       SegmentuaSortu (gValX[i], gValY[i], gValX[i]+gPaso, gValY[i],        0.000000);
129        if (PuntuHauExistitzenDa (gValX[i]+gPaso, gValY[i]+gPaso)) SegmentuaSortu (gValX[i], gValY[i], gValX[i]+gPaso, gValY[i]+gPaso,  45.000000);
130        if (PuntuHauExistitzenDa (gValX[i],       gValY[i]+gPaso)) SegmentuaSortu (gValX[i], gValY[i], gValX[i],       gValY[i]+gPaso,  90.000000);
```

```
131         if (PuntuHauExistitzenDa (gValX[i]-gPaso, gValY[i]+gPaso)) SegmentuaSortu (gValX[i], gValY[i], gValX[i]-gPaso, gValY[i]+gPaso, 135.000000);
132         if (PuntuHauExistitzenDa (gValX[i]-gPaso, gValY[i]))       SegmentuaSortu (gValX[i], gValY[i], gValX[i]-gPaso, gValY[i],       180.000000);
133         if (PuntuHauExistitzenDa (gValX[i]-gPaso, gValY[i]-gPaso)) SegmentuaSortu (gValX[i], gValY[i], gValX[i]-gPaso, gValY[i]-gPaso, 225.000000);
134         if (PuntuHauExistitzenDa (gValX[i],       gValY[i]-gPaso)) SegmentuaSortu (gValX[i], gValY[i], gValX[i],       gValY[i]-gPaso, 270.000000);
135         if (PuntuHauExistitzenDa (gValX[i]+gPaso, gValY[i]-gPaso)) SegmentuaSortu (gValX[i], gValY[i], gValX[i]+gPaso, gValY[i]-gPaso, 315.000000);
136     }
137     printf("Sortutako segmentu kopurua = %d\n", gCntSeg);
138 }
139
140 void SegmentuenOptimizazioa ()
141 {
142     bool cambiado = false;
143     int  i = 0, j = 0;
144     float XFin, YFin, angulo;
145
146     printf("Segmentuen optimizazioa\n");
147     while (i < gCntSeg){
148         cambiado = false;
149         XFin     = gSegXFin[i];
150         YFin     = gSegYFin[i];
151         angulo   = gSegAng[i];
152         j        = 0;
153         while ((!cambiado) && (j < gCntSeg)){
154             if ((i != j) && (XFin == gSegXIni[j]) && (YFin == gSegYIni[j]) && (gSegAng[j] > -0.500000) && (angulo == gSegAng[j])) {
155                 gSegXFin[i] = gSegXFin[j];
156                 gSegYFin[i] = gSegYFin[j];
157                 gSegAng[j]  = -0.500000;
158                 cambiado = true;
```

```
159             }
160             j++;
161         }
162         if (!cambiado) i++;
163     }
164     j=0;
165     for (i=0; i<gCntSeg; i++) {
166         if (gSegAng[i] > -0.5) j++;
167     }
168     printf("Sortutako segmentu kopurua = %d\n", j);
169 }
170
171 void DatFitxategiarenSorrera ()
172 {
173     FILE* fichero;
174     char nomFchDat [80] = "";
175
176     strcat(nomFchDat, gNomFch);
177     strcat(nomFchDat, ".dat");
178
179     printf("Segmentuen datuak dituen fitxategia sortzen \"%s\"\n", nomFchDat);
180     fichero = fopen(nomFchDat, "w");
181     fprintf(fichero,"  XIni,YIni,XFin,YFin\n");
182     for (int i=0; i<gCntSeg; i++) {
183         if (gSegAng[i] > -0.500000) fprintf(fichero,"  %f,%f,%f,%f,%f\n", gSegXIni[i], gSegYIni[i], gSegXFin[i], gSegYFin[i], gSegAng[i]);
184     }
185     fclose(fichero);
```

```
186 }
187
188 void DxfFitxategiarenSorrera ()
189 {
190     FILE* fichero;
191     char nomFchDxf [80] = "";
192
193     strcat(nomFchDxf, gNomFch);
194     strcat(nomFchDxf, ".dxf");
195
196     printf("DXF fitxategia sortzen \"%s\"\n", nomFchDxf);
197     fichero = fopen(nomFchDxf, "w");
198     fprintf(fichero,"0\n");
199     fprintf(fichero,"SECTION\n");
200     fprintf(fichero,"2\n");
201     fprintf(fichero,"ENTITIES\n");
202     fprintf(fichero,"0\n");
203     for (int i=0; i<gCntSeg; i++) {
204         if (gSegAng[i] > -0.500000) {
205             fprintf(fichero,"LINE\n");
206             fprintf(fichero,"8\n");
207             fprintf(fichero,"0\n");
208             fprintf(fichero,"10\n");
209             fprintf(fichero,"%f\n", gSegXIni[i]);
210             fprintf(fichero,"20\n");
211             fprintf(fichero,"%f\n", gSegYIni[i]);
212             fprintf(fichero,"11\n");
```

```
213             fprintf(fichero,"%f\n", gSegXFin[i]);
214             fprintf(fichero,"21\n");
215             fprintf(fichero,"%f\n", gSegYFin[i]);
216             fprintf(fichero,"0\n");
217         }
218     }
219     fprintf(fichero,"ENDSEC\n");
220     fprintf(fichero,"0\n");
221     fprintf(fichero,"EOF\n");
222     fclose(fichero);
223 }
224
225
226 int main(){
227     SarrerakoPuntuakHasieratu ();
228     DatuenFitxategiaIrakurri();
229     PuntuKopuruaZenbatu();
230     PuntuenArtekoPausuaZehaztu ();
231     SegmentuenSorrera();
232     SegmentuenOptimizazioa ();
233     DatFitxategiarenSorrera();
234     DxfFitxategiarenSorrera();
235
236     printf("Prozesua Amaitu Da\n");
237     system ("pause");
238     return 0;
239 }
240
```