(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) **World Intellectual Property Organization**
International Bureau

(43) **International Publication Date**
23 August 2018 (23.08.2018)

W!PO | PCT

(10) **International Publication Number**
WO 2018/151612 Al

(51) **International Patent Classification:**
G06T 15/04 (201 1.01) *G06T 17/00* (2006.01)

(21) **International Application Number:**
PCT/NZ20 18/0500 13

(22) **International Filing Date:**
19 February 2018 (19.02.2018)

(25) **Filing Language:** English

(26) **Publication Language:** English

(30) **Priority Data:**
20179005 14 17 February 2017 (17.02.2017) AU

(71) **Applicant: QUIVERVISION LIMITED** [NZ/NZ]; c/o Deloitte, 80 Queen Street, Auckland Central, Auckland 1010 (NZ).

(72) **Inventor: CLARK, Adrian;** c/o AJ Park, State Insurance Tower, 1 Willis Street, PO Box 949, Wellington (NZ).

(74) **Agent: AJ PARK;** Level 22, State Insurance Tower, 1 Willis Street, Wellington (NZ).

(81) **Designated States** *(unless otherwise indicated, for every kind cf national protection available):* AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(54) **Title:** TEXTURE MAPPING SYSTEM AND METHOD

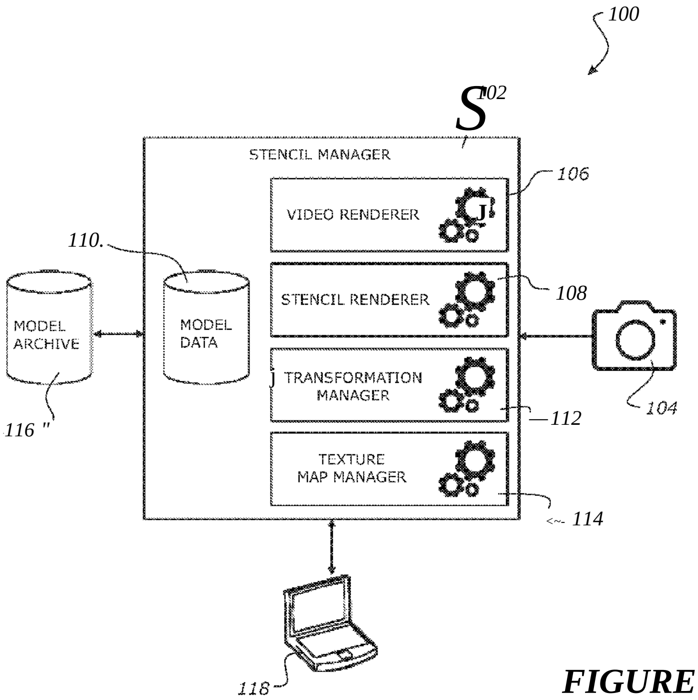

FIGURE 1

(57) **Abstract:** An aspect of the invention provides a texture mapping system configured to map texture to a three-dimensional model. The system includes a video renderer configured to render captured image data on a display to create rendered image data. The system includes a stencil renderer configured to generate at least one stencil from at least one model stored in a model data store, and render the at least one stencil on the display to create at least one rendered stencil. The system includes a transformation manager configured to map a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture, and map the at least one active region to at least part of a texture map associated to the at least one model. The system includes a texture map manager configured to copy the texture associated to the rendered stencil to the at least part of the texture map mapped to the

WO 2018/151612 A1

[Continued on nextpage]

(84) **Designated States** *(unless otherwise indicated, for every kind cf regional protection available):* ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *in black and white; the international application as filed contained color or greyscale and is available for download from PATENTSCOPE*

at least one active region.

TEXTURE MAPPING SYSTEM AND METHOD

FIELD OF THE INVENTION

This invention relates to improvements in respect of mapping colour and/or texture in image data generation, such as mapping colour and/or texture to 3D models.

BACKGROUND OF THE INVENTION

Various known digital media systems involve coloured or textured objects or characters. Often these systems have particular colours or textures which require significant time or significant expertise for a creator of the object to generate. It can be a particular operational challenge and overhead to find operators with expertise to both create the colour, texture and to map these to an object or character.

The time required for creating colour and/or texture for characters and objects can become greater where digital media is generated using 3D models. Traditional texture mapping involves an artist manually mapping how a two-dimensional (2D) texture maps onto a three-dimensional (3D) model. Often the view, orientation or configuration of the 3D model changes through a digital media presentation, requiring colour and/or texture content to be created for multiple viewpoints of the object or character.

There have been some attempts to at least partially automate texture mapping. One such technique is known as 'unfolding'. A 3D model is flattened out onto a 2D image in an arbitrary layout. An artist can then apply texture to the flattened image to texture one or more faces of the 3D model. Another technique is known as 'projection'. A 3D model is projected to a 2D image from the view of a known virtual camera. The artist can then fill in the texture from that view.

These known techniques have the potential to reduce some of the burden on an artist through automation. However, they do not permit an artist to texture and re-texture a 3D model in real time from a live video feed.

It would be of advantage to have a process which allows a user to capture colour and/or texture for generated image media.

It would be of advantage to have a process which allows a user to map a colour and/or texture for a 3D model.

It would be of particular advantage to have a process which allows a user to capture colour and/or texture for 3D models used in the generation of digital media.

It is an object of at least preferred embodiments of the present invention to address some of the aforementioned disadvantages. An additional or alternative object is to at least provide the public with a useful choice.

SUMMARY OF THE INVENTION

In accordance with an aspect of the invention, a texture mapping system is configured to map texture to a three-dimensional model. The system comprises a video renderer configured to render captured image data on a display to create rendered image data; a stencil renderer configured to: generate at least one stencil from at least one model stored in a model data store, and render the at least one stencil on the display to create at least one rendered stencil; a transformation manager configured to: map a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture, and map the at least one active region to at least part of a texture map associated to the at least one model; and a texture map manager configured to copy the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

The term 'comprising' as used in this specification means 'consisting at least in part of. When interpreting each statement in this specification that includes the term 'comprising', features other than that or those prefaced by the term may also be present. Related terms such as 'comprise' and 'comprises' are to be interpreted in the same manner.

In an embodiment the texture map is associated to a first view of the model that faces a user and a second view of the model that faces away from a user.

In an embodiment the texture map manager is configured to copy the texture associated to the rendered stencil to a part of the texture map associated only to the first view of the model.

In an embodiment the texture map manager is configured to copy the texture associated to the rendered stencil to both the part of the texture map associated to the first view of the model and the part of the texture map associated to the second view of the model.

In an embodiment the texture map manager is configured to copy the texture associated to a second stencil to a part of the texture map associated only to the second view of the model.

In an embodiment the stencil renderer is configured to accept user input so that the generated at least one stencil is at least partially user-defined.

In an embodiment the texture map manager is configured to generate the texture map.

In an embodiment the texture map manager is configured to generate a mapping between the generated texture map and the model.

In an embodiment the texture map manager is configured to retrieve the texture map from a storage device.

In accordance with a further aspect of the invention, a method of mapping texture to a three-dimensional model comprises rendering captured image data on a display to create rendered image data; generating at least one stencil from at least one model stored in a model data store; rendering the at least one stencil on the display to create at least one rendered stencil; mapping a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture; mapping the at least one active region to at least part of a texture map associated to the at least one model; and copying the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

In an embodiment the method further comprises associating the texture map to a first view of the model that faces a user and a second view of the model that faces away from a user.

In an embodiment the method further comprises copying the texture associated to the rendered stencil to a part of the texture map associated only to the first view of the model.

In an embodiment the method further comprises copying the texture associated to the rendered stencil to both the part of the texture map associated to the first view of the model and the part of the texture map associated to the second view of the model.

In an embodiment the method further comprises copying the texture associated to a second stencil to a part of the texture map associated only to the second view of the model.

In an embodiment the method further comprises accepting user input so that the generated at least one stencil is at least partially user-defined.

In an embodiment the method further comprises generating the texture map.

In an embodiment the method further comprises generating a mapping between the generated texture map and the model.

In an embodiment the method further comprises retrieving the texture map from a storage device.

In accordance with a further aspect of the invention, a system for mapping texture to a three-dimensional model comprises a storage device; a display; and a processor. The processor is programmed to: render captured image data on a display to create rendered image data, generate at least one stencil from at least one model stored in a model data store, render the at least one stencil on the display to create at least one rendered stencil, map a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture, map the at least one active region to at least part of a texture map associated to the at least one model, and copy the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

In accordance with a further aspect of the invention a computer-readable medium has stored thereon computer-executable instructions that, when executed by a processor, cause the processor to perform a method of mapping texture to a three-dimensional model. The method comprises rendering captured image data on a display to create rendered image data; generating at least one stencil from at least one model stored in a model data store; rendering the at least one stencil on the display to create at least one rendered stencil; mapping a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture; mapping the at least one active region to at least part of a texture map associated to the at least one model; and copying the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

Disclosed herein is a process operable to map under control of an operator colour and/or texture to a 3D model comprising overlaying on captured image data and/or video data stencil data carrying information on a stencil, capturing a selection of a part of the stencil which maps to a part of the model, capturing image data and/or video data overlaid by the selected part of the stencil and mapping the captured image data and/or video data to the 3D model.

Disclosed herein is a system operable to map under control of an operator colour and/or texture to a 3D model comprising a user interface operable to overlay on captured image data and/or captured video data stencil data carrying information on a stencil, and to receive selection inputs operable to select a part of the stencil which maps to a part of the model, and image capture module operable to capture image data and/or video data overlaid by the selected part of the stencil and a mapping module the captured image data and/or video data to the 3D model.

The image capture module may be operable to generate 2D mapping data carrying information on colour and/or texture captured background image and carrying information on the stencil. The system may be operable to display the mapping data.

Disclosed herein is a process operable to map under control of an operator colour and/or texture to a 3D model comprising overlaying on captured image data and/or video data stencil data carrying information on a stencil, wherein one or more parts of the stencil maps to a part of the model, capturing image data and/or video data overlaid by the stencil data and mapping the captured image data and/or video data to the 3D model.

The process may comprise storing mapping image data carrying information on texture and/or colour captured in captured image data, and using a defined mapping relationship between the mapping image data and the 3D model. The mapping image data may comprise texture and/or colour captured from multiple alternative image data and/or video data captured. The process may comprise displaying at a user interface the mapping image data.

Disclosed herein is a process of mapping under the control of an operator colour and/or texture to a 3D model comprising: generating a user interface which displays a stencil image data comprising background image data captured using a camera dependent on operator control inputs and which displays stored stencil data; and mapping captured background image data dependent on stored mapping data defining a mapping from data the stencil to the 3D model.

The operator control inputs may comprise a selection of a mask matching a component of the stencil.

The mask may define which pixels in the stencil image data are to be captured.

The mask may define a multiple stencil components.

This may allow stencil image data to carry information on a context for the operator while the mask defines the pixels to be captured for mapping to the 3D model. Selection of a mask may allow an operator to select a component.

Disclosed herein is a process for generating image data comprising a 3D model having mapped thereon under the control of an operator colour or texture, the process comprising: generating a first user interface displaying stencil data defining one more stencils wherein the stencil has a defined mapping to the model; capturing image data carrying information on an image to use for colour or texture for the model; displaying at the user interface one or more masks each defining one of the one or more components of the stencil; receiving mask selection data at the user interface to allow the operator to select a mask to select a capture region of the stencil for colour or texture mapping; receiving at the user interface data defining a capture input made by the operator; capturing image data within the capture region dependent on the received capture input; and mapping the captured background image data to the 3D image associated with the stencil selected.

The stencil data may be stored and retrieved in a format carrying information on a 2D outline.

The stencil data may be generated from a mesh of the 3D model. This may comprise determining a dot product of faces of the mesh and a viewing plane.

The process may comprise displaying one or more masks, each having a transparent part corresponding to a selected part of the stencil.

The process may comprise receiving input data to allow the operator to manipulate the stencil with respect to the background image data.

The background image data may comprise a frame in a video feed.

The background image data may comprise a stored image or video.

The stencil may be a 2D stencil.

The stencil may be a 3D stencil.

Disclosed herein is a system for mapping under control of an operator colour and/or texture to a 3D model, the system comprising: a first interface module operable to overlay stencil data on captured image data and/or video data, a second interface module operable to receive inputs defining a selection of a part of the stencil which maps to a part of the model, a capture module operable to capture image data and/or video data overlaid by the selected part of the stencil; and a mapping module operable to map the captured image data and/or video data to the 3D model.

The mapping module may be operable to map the captured image and/or video data dependent on a defined mapping between 2D mapping image data and the 3D model.

The mapping image data may comprise the stencil data which defines one or more regions of an image and comprise captured image and/or video data captured in one or more of the regions defined by the stencil data to display captured image and/or video data captured under control of the operator for mapping to meshes of the 3D model associated by defined mappings to components of the stencil.

The system may comprise a third interface module operable to display the mapping image data.

The selection of a part of the stencil may comprise selection from two or more masks for the stencil, each mask masking a captured image. Each mask may define a region of a stencil, or component of a stencil, which should define an area of capture of colour from a background image.

Disclosed herein is a system for mapping under the control of an operator colour and/or texture to a 3D model comprising: a user interface module operable to display background image data captured using a camera dependent on first operator control inputs and display stored stencil data carrying information on the outline of a set of stencil parts, each part having a data association with data carrying information on a part of the 3D model, and a mapping module operable to map this captured background image data dependent on the stored data association for a part of the stencil selected by second operator control inputs.

The second operator control inputs may comprise a selection of a mask matching a part of the stencil.

Disclosed herein is a system for generating image data comprising a 3D model formed of series of shapes arranged in a mesh having mapped thereon under the control of an operator colour or texture, the system comprising: a user interface module operable to display stencil data carrying information defining one more stencils defining one or more regions to represent parts of a stencil for which to capture image data; stored mapping data carrying information on a mapping for each stencil, the mapping being from mapping data comprising a stencil and mapping to mesh shapes of the model; a capture module capturing image data carrying information on an image to use for colour or texture for the model; a masking module operable to retrieve mask data carrying information on one or more masks each having one or more regions defining one of the one or more parts of the stencil; a mapping module operable to map the captured background image data to a part of the 3D image associated with the stencil selected, the mapping being dependent on a stored mapping from the stencil data to the mesh; and wherein the user interface is further operable to receive capture control inputs to allow the operator to initiate capture of image data within the capture region dependent on the received capture control input.

Disclosed herein is the display of stencil data carrying information on a stencil in a user interface to provide an operator with a reference in capturing background data wherein the stencil is included in a texture mapping image which has a defined mapping to a model.

Disclosed herein is a process of mapping under control of an operator colour and/or texture to a 3D model comprising overlaying stencil data on a captured image data and/or video data, capturing a selection of a part of the stencil which maps to a part of the model, capturing image data and/or video data overlaid by the selected part of the stencil and mapping the captured image data and/or video data to the 3D model.

As used here in the terms "first" and "second" are intended as broad designations without an implication of an order in a sequence. For example, a "first control input" may occur after a "second control input".

The invention in one aspect comprises several steps. The relation of one or more of such steps with respect to each of the others, the apparatus embodying features of construction, and combinations of elements and arrangement of parts that are adapted to affect such steps, are all exemplified in the following detailed disclosure.

To those skilled in the art to which the invention relates, many changes in construction and widely differing embodiments and applications of the invention will suggest themselves without departing from the scope of the invention as defined in the appended claims. The disclosures and the descriptions herein are purely illustrative and are not intended to be in any sense limiting. Where specific integers are mentioned herein which have known equivalents in the art to which this invention relates, such known equivalents are deemed to be incorporated herein as if individually set forth.

In addition, where features or aspects of the invention are described in terms of Markush groups, those persons skilled in the art will appreciate that the invention is also thereby described in terms of any individual member or subgroup of members of the Markush group.

As used herein, '(s)' following a noun means the plural and/or singular forms of the noun.

As used herein, the term 'and/or' means 'and' or 'or' or both.

As used herein, the term 'texture' includes 'colour'.

As used herein, the term 'image data' is associated to one or more of an image, a plurality of images, a sequence of images, video.

It is intended that reference to a range of numbers disclosed herein (for example, 1 to 10) also incorporates reference to all rational numbers within that range (for example, 1, 1.1, 2, 3, 3.9, 4, 5, 6, 6.5, 7, 8, 9, and 10) and also any range of rational numbers within that range (for example, 2 to 8, 1.5 to 5.5, and 3.1 to 4.7) and, therefore, all sub-ranges of all ranges expressly disclosed herein are hereby expressly disclosed. These are only examples of what is specifically intended and all possible combinations of numerical values between the lowest value and the highest value enumerated are to be considered to be expressly stated in this application in a similar manner.

The term 'computer-readable medium' should be taken to include a single medium or multiple media. Examples of multiple media include a centralised or distributed database and/or associated caches. These multiple media store the one or more sets of computer executable instructions. The term 'computer readable medium' should also be taken to include any medium that is capable of storing, encoding or carrying a set of instructions for execution by a processor and that cause the processor to perform any one or more of the methods described above. The computer-readable medium is also capable of storing, encoding or carrying data structures used by or associated with these sets of instructions. The term 'computer-readable medium' includes solid-state memories, optical media and magnetic media.

The terms 'component', 'module', 'system', 'interface', and/or the like as used in this specification in relation to a processor are generally intended to refer to a computer-related entity, either hardware, a combination of hardware and software, software, or software in execution. For example, a component may be, but is not limited to being, a process running on a processor, a processor, an object, an executable, a thread of execution, a program, and/or a computer. By way of illustration, both an application running on a controller and the controller can be a component. One or more components may reside within a process and/or thread of execution and a component may be localized on one computer and/or distributed between two or more computers.

The term 'connected to' as used in this specification in relation to data or signal transfer includes all direct or indirect types of communication, including wired and wireless, via a cellular network, via a data bus, or any other computer structure. It is envisaged that they may be intervening elements between the connected integers. Variants such as 'in communication with', 'joined to', and 'attached to' are to be interpreted in a similar manner. Related terms such as 'connecting' and 'in connection with' are to be interpreted in the same manner.

In this specification where reference has been made to patent specifications, other external documents, or other sources of information, this is generally for the purpose of providing a context for discussing the features of the invention. Unless specifically stated otherwise, reference to such external documents or such sources of information is not to be construed as an admission that such documents or such sources of information, in any jurisdiction, are prior art or form part of the common general knowledge in the art.

In the description in this specification reference may be made to subject matter which is not within the scope of the appended claims. That subject matter should be readily identifiable by a person skilled in the art and may assist in putting into practice the invention as defined in the presently appended claims.

Although the present invention is broadly as defined above, those persons skilled in the art will appreciate that the invention is not limited thereto and that the invention also includes embodiments of which the following description gives examples.

BRIEF DESCRIPTION OF THE DRAWINGS

Additional and further aspects of the present invention will be apparent to the reader from the following description of embodiments, given in by way of example only, with reference to the accompanying drawings in which:

Figure 1 shows a high level example of a texture mapping system;

Figure 2 shows an example of a method for mapping texture using a stencil generated from a 3D model;

Figures 3 to 8 show an example of applying the method of figure 2;

Figure 9 shows a further example of a method for mapping texture using a stencil generated from a 3D model;

Figures 10 to 15 show an example of applying the method of figure 9;

Figure 16 shows a further example of a method for mapping texture to a 3D model using a stencil generated from a 3D model;

Figure 17 shows a further example of a method for mapping texture using a stencil generated from a 3D model;

Figures 18-20 illustrate a model, the corresponding mesh, and texture mapping image of an embodiment of the present invention;

Figures 21 to 24 illustrate a stencil image and stencil mask, and the capture and mapping of colour;

Figure 25 illustrates the capture and mapping of colour process according to the same embodiment as figures 21 to 24;

Figures 26 to 28 illustrate the role of stencil and mask according to another embodiment of the present invention;

Figures 29 to 31 illustrate the role of a texture mapping image according to the same embodiment as figures 26 to 28;

Figures 32 to 34 illustrate the role of a tiled texture mapping image according to a further embodiment of the present invention;

Figure 35 illustrates an a process according to the same embodiment as figures 32 to 34;

Figures 36 to 38 illustrate a model with multiple components and a generated stencil image according to a yet further embodiment of the present invention;

Figures 39 to 41 illustrate mapping of colour and texture of a face to a model according to another embodiment of the invention; and

Figure 42 shows an example of a computing device 118 from figure 1.

Further aspects of the invention will become apparent from the following description of the invention which is given by way of example only of particular embodiments.

DETAILED DESCRIPTION

Figure 1 shows a high level example of a texture mapping system 100. The system 100 includes a stencil manager 102. The operation of the stencil manager 102 is described in more detail below.

In an embodiment the stencil manager 102 obtains image data, for example video background, that is captured by an image capture device for example a camera 104. The stencil manager is configured to obtain image data in the form of video via a live feed from camera 104. Alternatively or additionally the stencil manager obtains a data file representing recorded video from the camera 104.

In an embodiment a video renderer 106 is configured to render video background captured by the camera 104.

In an embodiment a stencil renderer 108 obtains at least one stencil from a plurality of stencils stored in model data store 110. In an embodiment the stencil renderer 108 generates at least one stencil from at least one model stored in the model data store 110. In an embodiment a stencil is created by the user. In an embodiment a stencil represents some component of a 3D model for which a user wishes to change a texture.

The stencil renderer 108 renders the retrieved and/or generated at least one stencil to an image display device for example a screen. In an embodiment the at least one stencil is obtained from an existing two-dimensional (2D) map in the model data store 110. In an embodiment the at least one stencil is generated from at least one three-dimensional (3D) model stored in the model data store 110.

In an embodiment a transformation manager 112 is configured to determine the position of the at least one stencil on the screen.

The position of the at least one stencil, in an embodiment, is passed to a texture map manager 114. The texture map manager 114 captures the relevant parts of the video background. In an embodiment the video is captured into an existing texture map. In an embodiment the video is captured into a new texture map.

In an embodiment an external model archive 116 is connected to the model data store 110. In an embodiment at least some of the 2D maps, 3D models and texture mapping information is exchanged between the model data store 110 and the external model archive 116. In an embodiment the model archive 116 is implemented as one or more of a file system on a disk, a service running on an external machine.

In an embodiment the stencil manager 102 is connected to a client device 118. In an embodiment the stencil manager 102 is implemented on the client device 118. In an embodiment the stencil manager 102 is implemented on at least one server connected to the client device 118.

Figure 2 shows an example of a method for mapping colour or texture using a stencil generated from a 3D model. More specifically the stencil is generated from an automatically created projection generated from an arbitrarily oriented model. In an embodiment the textures are automatically captured from a live video feed rather than being defined by a user.

In an embodiment the method copies a texture into an existing 2D texture map image. In an embodiment the 2D texture map image has been created for the 3D model by an artist in advance.

At step 200 the system finds corners of an image or live camera stream in screen coordinates.

At step 202 the system calculates mapping Ml which maps screen coordinates to image or live camera stream corners.

At step 204 the system finds corners of a 2D stencil generated from a 3D model in screen coordinates.

At step 206 the system calculates a mapping M2 which maps stencil coordinates to screen coordinates.

At step 208 the system initiates a loop through each triangle T in the 3D model component being stencilled.

At step 210 the system calculates the position of each vertex v[l,2,3] in T in the stencil image by projecting the vertex with the virtual camera.

At step 212 for each v the system calculates a position v_s in screen coordinates using M2.

At step 214 for each v_s, the system calculates the position v_i in background image/live camera stream coordinates using **Mi**.

At step 216 each pixel p_i in the background image/live camera stream contained within the triangle defined by v_i[l,2,3] is copied into the texture mapped image.

The method shown in figure 2 enables more than one view of the 3D model to be textured at the same time. Examples of view of the 3D model include a view facing the user and a view facing away. In an embodiment a 3D geometry of the model is projected to determine which elements of the screen should be copied to the texture map.

Figures 3 to 8 show an example of applying the method of figure 2. The figures show one embodiment of 3D Stencils. In this embodiment, colours and textures are copied from the background image into the model's existing texture map using the mapping defined by the 3D stencil.

Figure 3 shows a 3D model 300 before a stencil is generated. In an embodiment the data representing the 3D model is stored in the model data store 110.

Figure 4 shows an example of a texture map 400 of the same character as modelled in Figure 3. The texture map 400 is shown using a projection technique. The model 300 is shown projected to a first 2D image 402 and a second 2D image 404. First image 402 and second image 404 are projected from two different views of a virtual camera, or two different virtual cameras.

The texture map 400 is associated to the model 300. In an embodiment the association is performed by storing the texture map 400 in the model data store 110 together with a mapping between the texture map 400 and the model 300. Changes to the texture map 400 are mapped to changes to the model 300 as the texture map 400 and the model 300 are associated. It will be appreciated that the stored mapping is one example of an association. In an embodiment the association includes a 3D model having a related 2D texture map. In an embodiment the 2D texture map includes a projection 2D texture map and/or a flattened 2D texture map.

Figure 5 shows an example of a stencil 500. In an embodiment the stencil renderer 108 generates the stencil 500 from the model 300. The stencil renderer 108 renders the stencil 500 on a display to create what is shown in figure 5 as a rendered stencil.

Figure 6 shows the stencil 500 from figure 5. Figure 6 further shows a background image 602 or camera stream with which the user wishes to provide texture to the 3D model 300.

The rendered stencil has associated to it a plurality of coordinates. These coordinates represent the position of the rendered stencil on the display. The rendered background image data 602 also has associated to it a plurality of coordinates.

In an embodiment the transformation manager 112 from figure 1 is configured to map the coordinates associated to the rendered stencil to the coordinates for the rendered background data. The transformation manager 112 effectively identifies the position of the rendered stencil on the display relative to the rendered background image data.

Figure 7 shows a resulting texture map 700 after the 3D stencil has been captured and the texture from the image or video feed has been copied into the texture map.

Figure 8 shows a 3D model 800 with mapped texture. The texture map 400 is associated to the 3D model 300. Therefore the changes to the texture map shown in figure 7 are mapped to the 3D model resulting in the 3D model shown in figure 8.

Figure 9 shows a further example of a method for mapping colour or texture using a stencil generated from a 3D model. More specifically the stencil is generated from an automatically created projection generated from an arbitrarily oriented model. In an embodiment the textures are automatically captured from a live video feed rather than being defined by a user.

In an embodiment the method copies a texture into an existing 2D texture map image. In an embodiment the 2D texture map image has been created for the 3D model by an artist in advance.

The method shown in figure 9 differs from the method shown in figure 2 in that only the view facing the user is textured at the time of capture. The method shows a technique for front-facing mapping colour or texture to a 3D model using the 3D model to generate a stencil.

Many of the steps of the method shown in figure 9 are described above with reference to the method shown in figure 2. The method includes an additional step shown at 900 of calculating a dot product of a face to determine whether or not it faces the user. Only if it faces the user does it perform a capture operation for that face.

Figures 10 to 15 show an example of applying the method of figure 9. In this embodiment, mapping is only done for triangles or polygons which face towards the screen, or front-facing mapping, compared to mapping to all triangles or polygons of a 3D model.

Figure 10 shows a 3D model 1000 prior to generation of a stencil or mapping of colour or texture. The model 1000 shown in figure 10 is similar to the model 300 shown in figure 3. In an embodiment the data representing the model 1000 is stored in the model data store 110.

In an embodiment, a texture map similar to the texture map 400 of figure 4 is associated to the model 1000. In an embodiment the association is performed by storing the texture map in the model data store 110 together with a mapping between the texture map and the model 300. Changes to the texture map are mapped to changes to the model 1000 as the texture map and the model 1000 are associated. It will be appreciated that the stored mapping is one example of an association. In an embodiment the association includes a 3D model having a related 2D texture map. In an embodiment the 2D texture map includes a projection 2D texture map and/or a flattened 2D texture map.

The stencil renderer 108 generates a stencil from the model 1000. In an embodiment the stencil is similar to stencil 500 from figure 5. The stencil renderer 108 renders the stencil 500 on a display to create what is shown in figure 5 as a rendered stencil.

Figure 11 shows the stencil 1100 generated from the model 1000 of Figure 10 being used in the capture of a background image or camera stream 1102. The background shown in figure 11 is similar to the background shown in figure 6.

As discussed above, the transformation manager 112 from figure 1 is configured to map the coordinates associated to the rendered stencil to the coordinates for the rendered background data. The transformation manager 112 effectively identifies the position of the rendered stencil on the display relative to the rendered background image data.

Figure 12 shows a 3D model 1200 to which colour has been mapped to all faces of the 3D model, such as including to the front-facing and rear-facing triangles within the mesh of the 3D model. In an embodiment a texture map associated to the model 1000 of figure 10 is updated by copying into it a texture from the image or video feed 1102 shown in figure 11. The texture map is associated to the 3D model 1000. Therefore the changes to the texture map are mapped to the 3D model resulting in the 3D model shown in figure 12.

Figure 12 shows a first view 1202 of the model 1000 that faces the user. Also shown is a second view 1204 that faces away from the user. As shown, a texture from the same background image or camera stream is applied to both the first view 1202 and the second view 1204.

Figure 13 shows a 3D model 1300 to which texture has been mapped to the first view 1202 of the model 1000 that faces the user. No texture has been mapped to a second view 1302 that faces away from the user. The texture from the background image or camera stream is not applied to both the first view 1202 and the second view 1302. The texture is applied to only one of first view 1202 and second view 1302. The second view 1302 remains untextured.

Figure 14 illustrates an embodiment of the method of figure 9. A first texture 1400 is captured for a front view of the mesh and a second texture 1402 is captured for a back view of the mesh. In an embodiment the first texture 1400 is different to the second texture 1402. The same algorithm used to capture different textures for the front and back views of the mesh.

Figure 15 shows different background images used to obtain the result shown in figure 14. The texture applied to a first view 1500 of the model is different to the texture applied to a second view 1502 of the model.

Figure 16 shows a further example of a method for mapping colour or texture to a 3D model using a stencil generated from a 3D model. More specifically the stencil is generated from an automatically created projection generated from an arbitrarily oriented model. In an embodiment the textures are automatically captured from a live video feed rather than being defined by a user.

In an embodiment the method creates a new 2D texture map. In an embodiment the method also calculates a mapping from the 3D model to the texture map image.

This can be contrasted with the methods described above with reference to figures 2 and 9 that copy a texture into an existing 2D texture map image.

The method shown in figure 16 has some steps in common with the method shown in figure 2. For example the first 4 steps 1602, 1604, 1606, and 1608 are equivalent to steps 200, 202, 204, and 206 respectively from figure 2.

At step 1602 the system finds corners of an image or live camera stream in screen coordinates.

At step 1604 the system calculates a mapping MI which maps screen coordinates to the corners of the image or live camera stream.

At step 1606 corners of a generated 2D stencil are found in screen coordinates.

At step 1608 the system calculates a mapping M2 which maps stencil coordinates to screen coordinates.

At step 1610 the system performs one of two steps. In an embodiment one step copies the entire background image or live camera feed into the model texture. In an embodiment one step calculates the bounds of the stencil and only copies that subsection of the background image or live camera feed into the model texture.

Steps 1612, 1614, and 1616 are equivalent to steps 208, 210, and 212 respectively from figure 2.

At step 1612 the system initiates a loop through 1614, 1616 and 1618 in order to loop through each triangle T in the 3D mesh part, or component, which is being stencilled.

At step 1614 the system calculates a position of each vertex v[l,2,3] in T in the stencil image by projecting the vertex with the virtual camera.

At step 1616, for each of v;, the system calculates the position vs in screen coordinates using M2.

At step 1618 the system stores the value of v_s as the texture map coordinate for vertex v.

The method shown in figure 16 is similar to the method shown in figure 2 in that it enables more than one view of the 3D model to be textured at the same time. Examples of views of the 3D model include a view facing the user and a view facing away. In an embodiment a 3D geometry of the model is projected to determine which elements of the screen should be copied to the texture map.

In this embodiment the method discards any existing texture mapping image, and instead uses the captured pixels as a texture map and adjusts the texture mapping for each triangle in the mesh.

The method operates for example on the 3D model 300 shown in figure 3. In an embodiment the data representing the 3D model is stored in the model data store 110.

In an embodiment the stencil Tenderer 108 generates a stencil from the model 300. An example of a stencil is shown at 500 in figure 5. The stencil renderer 108 renders the stencil 500 on a display to create what is shown in figure 5 as a rendered stencil.

A background image or camera stream is provided with which the user wishes to provide texture to the 3D model 300. An example of a background image is shown at 602 in figure 6.

The method creates a new texture map and associates the new texture map to the 3D model 300. The new texture map replaces the original texture map. The new texture map is mapped to the 3D model resulting in a 3D model similar to the 3D model shown in figure 8.

Figure 17 shows a further example of a method for mapping colour or texture using a stencil generated from a 3D model. More specifically the stencil is generated from an automatically created projection generated from an arbitrarily oriented model. In an embodiment the textures are automatically captured from a live video feed rather than being defined by a user.

In an embodiment the method creates a new texture map and associates the new texture map to the 3D model 300 for example. As described above with reference to figure 16 the new texture map is mapped to the 3D model resulting in a 3D model similar to the 3D model shown in figure 8.

The method shown in figure 17 differs from the method shown in figure 16 in that only the view facing the user is textured at the time of capture. The method shows a technique for front-facing mapping colour or texture to a 3D model using the 3D model to generate a stencil.

Many of the steps of the method shown in figure 17 are described above with reference to the method shown in figure 16. The method includes an additional step shown at 1700 of calculating a dot product of a face to determine whether or not it faces the user. Only if it faces the user does it perform a capture operation for that face.

In an embodiment the method shown in figure 17 includes determining whether the dot product of the triangle normal T_n and the camera view vector is less than 0.

An embodiment of a texture mapping system and process is described and illustrated with reference to Figures 18 to 20. Figure 18 shows a 3D model 1800 such as may be used in an augmented reality system according to an embodiment of the present invention. The character may be included in a game or in an augmented reality display or interface.

The 3D model of this example has been created by an artist. The 3D model is represented in a computer as a series of triangles or polygons, known as a "mesh" 1900, as shown in Figure 19. The triangles or polygons connect together to represent a 3 dimensional object.

Once the mesh 1900 has been created a process known as texture mapping is used. The artist creates a 2D texture mapping image 2000 and then for each triangle or polygon in the mesh 1900 they define a region of the 2D image for which the colour or texture or both should be applied to that particular triangle.

In practical terms an association between triangles or polygons and regions of the 2D image is stored by the system and used in the mapping. This allows captured colour or texture "texture" to be defined for each polygon based on a 2D texture mapping image.

Figure 21 shows a stencil 2100 according to an embodiment of the present invention. The stencil 2100 is displayed at a graphical user interface 2102. The stencil 2100 is a "2D stencil" which is a visual representation for the operator of regions of the model for which colour or texture would be captured and allows an operator to define which parts of a camera system or image to use to capture colour or texture for given regions of the 3D model. The stencil 2100 of this example represents a character which will be displayed by an augmented reality system.

The stencil 2100 in this example outlines parts of the character which might be given different colour or texture. **Figure 23** for example, shows the penguin represented with the following parts or components: a hat 2300, a face 2302, a beak 2304, a scarf 2306, an abdomen 2308, feet 2310 and a rear and lateral section 2312 including wings.

The user interface 2102 of this embodiment displays the stencil 2100 as overlaid on a background image 2104 captured by a camera (not shown) which communicates with a system (not shown). The system generates or presents the user interface 2102 with the stencil 2100 overlaying a background image 2104. In this example the image is one image of a stream of images forming video stream (not shown).

Figure 22 shows a mask 2200 which, in this example, has previously been selected at the user interface by the operator from a set of masks (not shown). The mask 2200 shown in Figure 22 is transparent in the region defined by the stencil to represent a component of the model 1800 and stencil 2100 for which colour or texture is to be captured and later mapped to a 3-D model.

In this example the mask 2200 is for the hat 2300 of the character. The mask 2200 is black in the rest of the frame. In this embodiment selection of the mask 2200 allows the operator to actually select a part of the stencil for which to capture colour and/or texture (not shown) from the background image 2104 for stencil 2100.

In the example shown the background image is of a hand which has a brown colour. The brown colour and hand texture is captured and masked to the outline of the hat 2300 by the mask 2200 and rendered to a masked stencil image 2304 so the hat 2300 of the 3D model 1800 is displayed as the colour of the hand in the background image 2104.

In this example a process using a stencil 2100 and different masks (not shown) can be used to render separately captured colour or texture to a common texture mapping image 2318. The texture mapping image in this embodiment has a defined mapping from pixels of the texture mapping image to triangles of the mesh 1900 of the model 1800.

Figure 23 shows a 2D texture mapped image with a hat 2300 which has the colour captured by the stencil 2100 from Figure 21, and with other components which have been coloured using alternative selected masks and capturing steps.

Data carrying the background image 2104 which was overlaid by the part of the stencil 2100 corresponding to a hat 2300 selected by selection of the mask 2200 at a time when the operator initiated capture of the background data has been mapped to the texture mapping image 2318.

The 2D texture mapping image 2318 of the character has been coloured with the background image 2104. The operator has controlled colour being added to the texture mapping image 2318 by manipulating the camera (not shown) of a device (not shown) displaying the stencil 2100 and background image 2104 to arrange a desired background image to be captured.

The operator has also controlled which captured background 2104 and colour being added to specific components of the texture mapping image 2318 by selection of the masks 2200 for given components. The operator also controls capture of the background image and initiation of mapping of the captured background image 2104 to texture mapping image 2318.

The 3D model 1800 with the same coloured hat is shown in a different pose and orientation to **Figure 24.** This illustrates that the component of the stencil 2100 for the hat 2300 maps colour captured from the background image of the hand shown in Figure 21 to one of the parts of the 3D model.

Figure 24 shows a 3D model 1800 which has a 3D hat part corresponding to the hat part 2300of the stencil 2100, as displayed at the interface 2102 and as included in the texture mapping image 2318. In this embodiment this mapping has been performed using a stored association between the hat part 2100 of the stencil 2 and the 3D hat part 2318.

A process illustrated with reference to Figures 22 to 24 will now be illustrated in reference to **Figure 25** which shows steps in a process. In the example of Figure 25 the background image may be part of a series of images forming a live camera stream.

At step 2502 the system finds corners of an image or live camera stream in coordinates of the screen of the user interface.

At step 2504 the system calculates mapping M 1 which maps screen coordinates to image/live camera stream corners.

At step 2506 the system finds corners of the 2D stencil in screen coordinates.

At step 2508 the system calculates mapping M to which maps stencil coordinates to screen coordinates.

At step 2510 the initiates a loop for 2512 to 2516.

At step 2512, if P is nonblack, the system calculates the position Ps in the screen coordinates using M2.

At step 2514 the position of Ps in the live image/live camera stream (PI) is calculated using Ml.

At step 2516 the pixel colour value at pi is copied to the texture image.

Figure 26 illustrates a 3D model 2600 according to another embodiment of the present invention. The model shown in Figure 26 is prior to mapping of colour from a stencil image shown in **Figure 27.**

Figure 27 shows a simple stencil 2700 in the form of a square, or tile, overlaid on a background image 2702.

Figure 28 shows a mask 2800 selected by inputs (not shown) of an operator at a user interface (not shown) by which an operator selected components of a 3D model to which the stencil image is to be mapped. The mask 2800 is an image with pixels rendered black other than in regions for a selected component of the 3D model, such as the dress in this example, to be coloured to mask those parts. In this example the mask defines a dress and head piece of the 3D model 2600.

Figure 29 shows the result of the mapping process in the form of a 3D model 2600 with colour and, in this example pattern or texture, mapped from the background image or camera stream shown in Figure 27.

In this example the colour and texture from the stencil 2700 has been mapped to the parts of the 3D model corresponding to the mask 2800. As will be apparent from the preceding illustration with reference to Figure 21 to 24, the colour and texture are mapped from the 2D texture mapping image 3100, shown in **Figure 31,** to the 3D model 2900shown in Figure 29. Figure 30 shows the same coloured and textured model from a different view point to Figure 29, illustrating that the model is 3D.

Figures 32 to 34 illustrate a further embodiment, where the captured stencil image has been shrunk and replicated in a 5 by 5 tile effect. Figure 32 shows a coloured and, in this example textured or patterned, 3D model. Figure 33 shows the same coloured and textured model from an alternative view point. Figure 34 shows a masked stencil image in which the captured stencil image shown in Figure 27 has been shrunk and replicated in a 5 by 5 tile effect.

The process illustrated with reference to Figure 32 to 34 will now be further illustrated with reference to a process shown in Figure 35. Many of the steps shown in figure 35 are similar to the steps shown in figure 25.

At step 2502 corners of an image or live camera stream are found in screen coordinates.

At step 2504 and mapping Ml is calculated. Ml maps screen coordinates to image or live camera stream corners.

At step 2506 corners of a 2D stencil are found in screen coordinates.

At step 2508 a mapping M2 is calculated. M2 maps stencil coordinates to screen coordinates.

At step 3500 the algorithm initiates a loop through steps 2510 to 2514 and 3502. The number of times (Ix, Iy) should be tiled horizontally (Th) and tiled vertically (Tv) is determined for this loop in the algorithm.

At step 2510 the algorithm initiates a loop through each pixel P in the stencil mask.

At step 2512 the algorithm determines if P is non-black and if so calculates its position Ps in screen coordinates using M2.

At step 2514 the algorithm calculates the position of Ps in the image or live camera stream (Pi) using Ml.

At step 3502 the pixel colour value at Pi is copied into a texture image at position (Pix/tw)*ix+ox (Piy/Th)*iy+oy, where ox and oy are x and y offsets.

In alternative embodiment of the present invention in which 3D stencil for use in place of 2D stencil will now be illustrated.

In the case of a 2D stencil, the stencil must be created by an author. A 3D stencil uses the hierarchy of meshes in a model to determine which components of the model should be used as a stencil. The meshes are rendered to mimic a 2D stencil.

A "virtual camera" is used to render the 3D geometry to a two dimensional plane, which is used as a stencil. The virtual camera can be used to "project" any point in 3D space to a 2D image, so it can be determined with any piece of geometry that appears in a 2D image.

Figure 36 shows a 3D model 3600, rendered as a mesh where the mesh is not visible as shown. Figure 37 shows a set of sub-models, sub-meshes, or components 3700 which make up the whole 3D model. Figure 38 shows the 3D model in which the part of the model for the hair 3800 of the character is rendered so as to provide a stencil. In this example this is achieved by rendering black the triangles of the mesh which have a dot product of 0 with the virtual camera plane. This generates a stencil with an outline defining a region, for hair in this particular example, for which a background in which is captured to provide cover or texture.

Because 3D stencils are generated using the mesh of the model, the model can be moved, rotated, resized or otherwise manipulated in 3 dimensions to similarly manipulate the stencil generated from the model. In this particular embodiment the system maps texture only to triangles or polygons of the mesh that are facing the virtual camera viewpoint screen. This is done by calculating the "normal" of the given triangle or polygon and calculating the dot product of this normal and a view vector of the virtual camera. If this is less than 0, the polygon is mapped. In alternative embodiments all triangles or polygons in the part of the mesh used to provide a stencil are mapped.

A process for mapping a captured image of a face to a model of a head, which is then added in this example to a 3-D character, will now be illustrated with reference to figures 39 to 41.

Figure 39 shows a stencil for a face area overlaid over a live camera stream including a face in the background to the stencil. **Figure 40,** shows similarly to as described above the colour and texture representing an image of the face within the stencil area is mapped to a 3D model of a head. In this example the mapping is an area in which a face would naturally appear on the 3D model.

Figure 41 shows a 3D model of a character to which has been added the 3D model of the head with colour and texture representing a face captured by the operator using the interface with stencil.

In some embodiments a graphical user interface allows a stencil to be manipulated by an operator, such as by translating, rotating, resizing or other manipulations known to the reader.

In some embodiments, the graphical user interface is generated and displayed by a device (not shown) which has a camera (not shown).

Some embodiments of the present invention capture images as image data, such as raster data or other forms, standards or protocols known to the reader.

In some embodiments texture which is captured using a stencil and mask can be manipulated by various known manipulations, such as shrinking and repeating, as illustrated with reference to figure 34, or such as translating.

In some embodiments of the present invention multiple stencils and multiple masks may be used for a single model. For example, in place of the stencil outlining a character with a hat, scarf and fur and multiple masks allowing the selection of captured colour or texture for each of these there may be used a stencil for each of the hat, scarf, fur and a mask for each of these.

As this will be apparent to the reader the system operating as illustrated with reference to figures 21 to 24, for example, allows an operator to control the application or mapping of colour and/or texture to parts of the 3D model by controlling initiation of the capture of a captured background overlaid by a stencil displayed by the system.

As this will be apparent to the reader the system operating as illustrated with reference to figures 21 to 24, for example, allows an operator to control the application or mapping of colour and/or texture to parts of the 3D model by manipulating the stencil.

As this will be apparent to the reader the system operating as illustrated with reference to figures 21 to 24, for example, allows an operator to control the application or mapping of colour and/or texture to parts of the 3D model by manipulating a camera used to capture background data to control the background image and/or video made available for capture and mapping to the 3D model.

As this will be apparent to the reader the system operating as illustrated with reference to figures 21 to 24, for example, allows an operator to control the application or mapping of colour and/or texture to parts of the 3D model by selecting a mask to determine which part of the stencil is used for the capture of background image colour and/or texture.

In various embodiments of the present invention provide displayed stencil data provides a means for guiding the operator to capture some video or image data which is then applied to the 3D model. The stencil image acts as a visual guide to give context of what is being captured from the image/video, and where it will be applied to the model. In various embodiments also algorithms determine how the parts of the image/video which are captured are applied to the model.

Some embodiments of the present invention capture video as video data, such as a video signal, stream or other video data forms, standards or protocols known to the reader.

In some embodiments a background image may be captured from a video stream prior to mapping from a stencil, or stencil image, to a 3D model.

In various embodiments the background image may be any bitmap data, raster data or other suitable forms of data known to the reader.

In various embodiments a texture mapping image is used as a predefined reference between a 2D representation of a model.

In various embodiments the texture mapping image has added colour and/or texture captured from a background image and/or video stream dependent on operator inputs at a user interface displaying a texture image comprising the background image and stencil data. This provides a reference for the operator controlling which parts of a background image are to be mapped to given components of the model displayed in the texture mapping image. In various embodiments data displayed texture image and the texture mapping image includes stencil data.

In some embodiments mapping process determine a projection of given vertices of the model onto the 2D texture mapping image.

In other embodiments a defined texture image is replaced with a texture mapping image generated from the texture image after background image data is captured.

Various embodiments implement the processes illustrated herein using a set of computer rules.

Various embodiments use stored texture mapping image template to which captured colour and/or texture is added. In some embodiments the texture mapping image template includes data carrying information on the same stencil as stencil data which is displayed at the user interface, overlaid on a background image.

In alternative embodiments a defined mapping may use a defined algorithm dependent on a texture mapping image template.

Figure 42 shows an embodiment of a suitable computing device 4200 to implement embodiments of one or more of the systems and methods disclosed above, such as the client device 118, the stencil manager 102, the video renderer 106, the stencil renderer 108, the transformation manager 112, the texture map manager 114, the model data 110, and/or model archive 116. The computing device of Figure 42 is an example of a suitable computing device. It is not intended to suggest any limitation as to the scope of use or functionality of the operating environment.

Example computing devices include, but are not limited to, personal computers, server computers, hand-held or laptop devices, mobile devices, multiprocessor systems, consumer electronics, mini computers, mainframe computers, and distributed computing environments that include any of the above systems or devices. Examples of mobile devices include mobile phones, tablets, and Personal Digital Assistants (PDAs).

Although not required, embodiments are described in the general context of 'computer readable instructions' being executed by one or more computing devices. In an embodiment, computer readable instructions are distributed via tangible computer readable media.

In an embodiment, computer readable instructions are implemented as program modules. Examples of program modules include functions, objects, Application

Programming Interfaces (APIs), and data structures that perform particular tasks or implement particular abstract data types. Typically, the functionality of the computer readable instructions is combined or distributed as desired in various environments.

Shown in figure 42 is a system 4200 comprising a computing device 4205 configured to implement one or more embodiments described above. In an embodiment, computing device 4205 includes at least one processing unit 4210 and memory 4215. Depending on the exact configuration and type of computing device, memory 4215 is volatile (such as RAM, for example), non-volatile (such as ROM, flash memory, etc., for example) or some combination of the two.

A server 4220 is shown by a dashed line notionally grouping processing unit 4210 and memory 4215 together.

In an embodiment, computing device 4205 includes additional features and/or functionality.

One example is removable and/or non-removable additional storage including, but not limited to, magnetic storage and optical storage. Such additional storage is illustrated in Figure 42 as storage 4225. In an embodiment, computer readable instructions to implement one or more embodiments provided herein are maintained in storage 4225. In an embodiment, storage 4225 stores other computer readable instructions to implement an operating system and/or an application program. Computer readable instructions are loaded into memory 4215 for execution by processing unit 4210, for example.

Memory 4215 and storage 4225 are examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, Digital Versatile Disks (DVDs) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by computing device 4205. Any such computer storage media may be part of device 4205.

In an embodiment, computing device 4205 includes at least one communication connection 4240 that allows device 4205 to communicate with other devices. The at least one communication connection 4240 includes one or more of a modem, a Network Interface Card (NIC), an integrated network interface, a radio frequency transmitter/ receiver, an infrared port, a USB connection, or other interfaces for connecting computing device 4205 to other computing devices.

In an embodiment, the communication connection(s) 4240 facilitate a wired connection, a wireless connection, or a combination of wired and wireless connections. Communication connection(s) 4240 transmit and/or receive communication media.

Communication media typically embodies computer readable instructions or other data in a "modulated data signal" such as a carrier wave or other transport mechanism and includes any information delivery media. The term "modulated data signal" includes a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal.

In an embodiment, device 4205 includes at least one input device 4245 such as keyboard, mouse, pen, voice input device, touch input device, infrared cameras, video input devices, and/or any other input device. In an embodiment, device 4205 includes at least one output device 4250 such as one or more displays, speakers, printers, and/or any other output device.

Input device(s) 4245 and output device(s) 4250 are connected to device 4205 via a wired connection, wireless connection, or any combination thereof. In an embodiment, an input device or an output device from another computing device is/are used as input device(s) 4245 or output device(s) 4250 for computing device 4205.

In an embodiment, components of computing device 4205 are connected by various interconnects, such as a bus. Such interconnects include one or more of a Peripheral Component Interconnect (PCI), such as PCI Express, a Universal Serial Bus (USB), firewire (IEEE 13104), and an optical bus structure. In an embodiment, components of computing device 4205 are interconnected by a network. For example, memory 4215 in an embodiment comprises multiple physical memory units located in different physical locations interconnected by a network.

It will be appreciated that storage devices used to store computer readable instructions may be distributed across a network. For example, in an embodiment, a computing device 4255 accessible via a network 4260 stores computer readable instructions to implement one or more embodiments provided herein. Computing device 4205 accesses computing device 4255 in an embodiment and downloads a part or all of the computer readable instructions for execution. Alternatively, computing device 4205 downloads portions of the computer readable instructions, as needed. In an embodiment, some instructions are executed at computing device 4205 and some at computing device 4255.

A client application 4285 enables a user experience and user interface. In an embodiment, the client application 4285 is provided as a thin client application configured to run within a web browser. The client application 4285 is shown in figure 42 associated to computing device 4255. It will be appreciated that application 4285 in an embodiment is associated to computing device 4205 or another computing device.

It is to be understood that the present invention is not limited to the embodiments described herein and further and additional embodiments within the spirit and scope of the invention will be apparent to the skilled reader from the examples illustrated with reference to the drawings. In particular, the invention may reside in any combination of features described herein, or may reside in alternative embodiments or combinations of these features with known equivalents to given features. Modifications and variations of the example embodiments of the invention discussed above will be apparent to those skilled in the art and may be made without departure of the scope of the invention as defined in the appended claims.

CLAIMS

1. A texture mapping system configured to map texture to a three-dimensional model, the system comprising:

a video renderer configured to render captured image data on a display to create rendered image data;

a stencil renderer configured to:

generate at least one stencil from at least one model stored in a model data store, and

render the at least one stencil on the display to create at least one rendered stencil;

a transformation manager configured to:

map a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture, and

map the at least one active region to at least part of a texture map associated to the at least one model; and

a texture map manager configured to copy the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

2. The system of claim 1 wherein the texture map is associated to a first view of the model that faces a user and a second view of the model that faces away from a user.

3. The system of claim 2 wherein the texture map manager is configured to copy the texture associated to the rendered stencil to a part of the texture map associated only to the first view of the model.

4. The system of claim 3 wherein the texture map manager is configured to copy the texture associated to the rendered stencil to both the part of the texture map associated to the first view of the model and the part of the texture map associated to the second view of the model.

5. The system of claim 3 wherein the texture map manager is configured to copy the texture associated to a second stencil to a part of the texture map associated only to the second view of the model.

6. The system of any one of the preceding claims wherein the stencil renderer is configured to accept user input so that the generated at least one stencil is at least partially user-defined.

7. The system of any one of the preceding claims wherein the texture map manager is configured to generate the texture map.

8. The system of claim 7 wherein the texture map manager is configured to generate a mapping between the generated texture map and the model.

9. The system of any one of claims 1 to 6 wherein the texture map manager is configured to retrieve the texture map from a storage device.

10. A method of mapping texture to a three-dimensional model, the method comprising:

rendering captured image data on a display to create rendered image data;

generating at least one stencil from at least one model stored in a model data store;

rendering the at least one stencil on the display to create at least one rendered stencil;

mapping a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture;

mapping the at least one active region to at least part of a texture map associated to the at least one model; and

copying the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

11. The method of claim 10 further comprising associating the texture map to a first view of the model that faces a user and a second view of the model that faces away from a user.

12. The method of claim 11 further comprising copying the texture associated to the rendered stencil to a part of the texture map associated only to the first view of the model.

13. The method of claim 12 further comprising copying the texture associated to the rendered stencil to both the part of the texture map associated to the first view of the model and the part of the texture map associated to the second view of the model.

14. The method of claim 12 further comprising copying the texture associated to a second stencil to a part of the texture map associated only to the second view of the model.

15. The method of any one of claims 10 to 14 further comprising accepting user input so that the generated at least one stencil is at least partially user-defined.

16. The method of any one of claims 10 to 15 further comprising generating the texture map.

17. The method of claim 16 further comprising generating a mapping between the generated texture map and the model.

18. The method of any one of claims 1 to 15 further comprising retrieving the texture map from a storage device.

19. A system for mapping texture to a three-dimensional model, the system comprising:

a storage device;

a display; and

a processor programmed to:

render captured image data on a display to create rendered image data,

generate at least one stencil from at least one model stored in a model data store,

render the at least one stencil on the display to create at least one rendered stencil,

map a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture,

map the at least one active region to at least part of a texture map associated to the at least one model, and

copy the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

20. A computer-readable medium having stored thereon computer-executable instructions that, when executed by a processor, cause the processor to perform a method of mapping texture to a three-dimensional model, the method comprising:

rendering captured image data on a display to create rendered image data;

generating at least one stencil from at least one model stored in a model data store;

rendering the at least one stencil on the display to create at least one rendered stencil;

mapping a plurality of coordinates on the display associated to the at least one rendered stencil to a plurality of coordinates on the display associated to the rendered image data to create at least one active region within the rendered stencil having an associated texture;

mapping the at least one active region to at least part of a texture map associated to the at least one model; and

copying the texture associated to the rendered stencil to the at least part of the texture map mapped to the at least one active region.

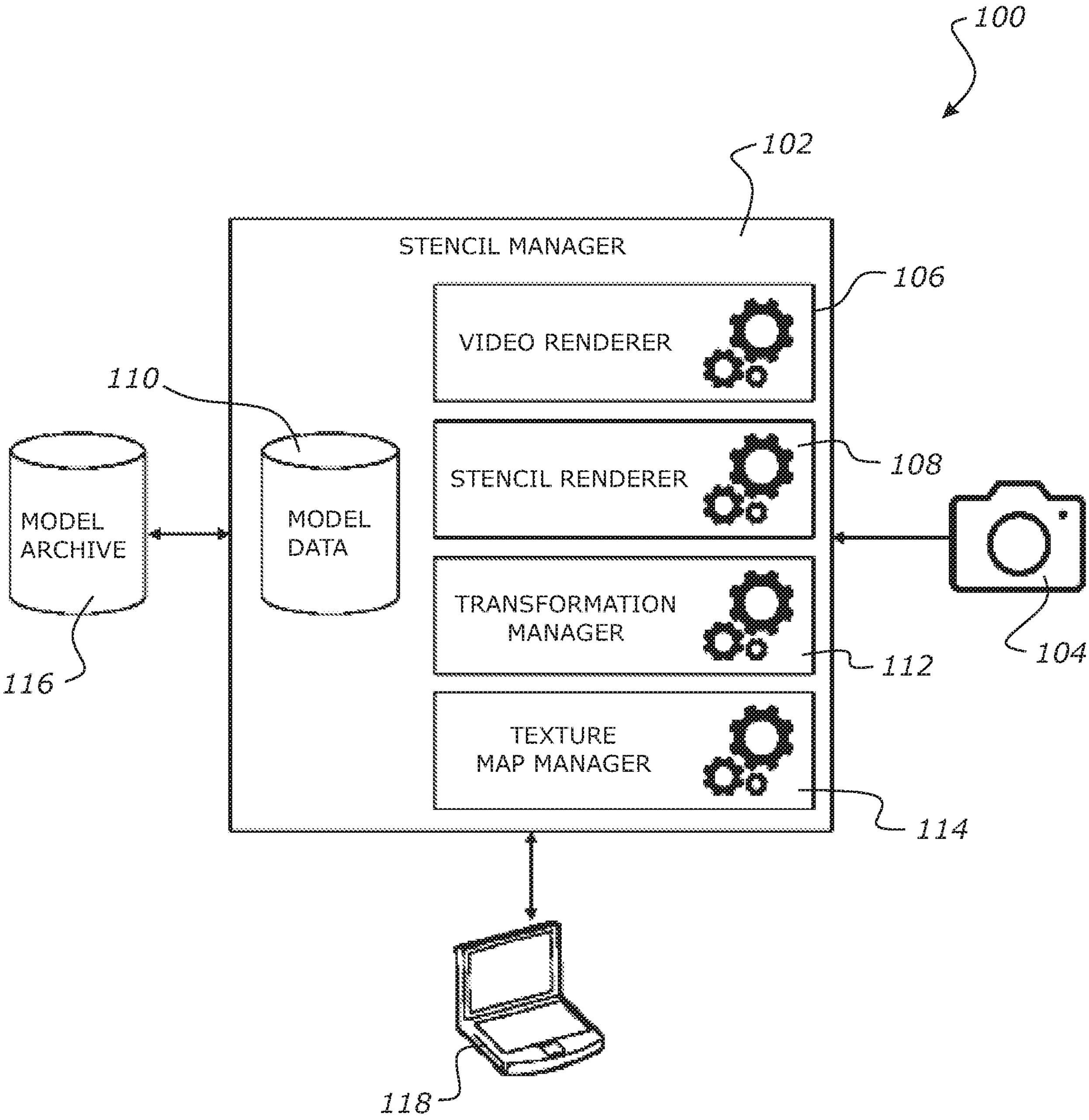
100
102
STENCIL MANAGER
106
VIDEO RENDERER
110
108
STENCIL RENDERER
MODEL ARCHIVE
MODEL DATA
TRANSFORMATION MANAGER
104
116
112
TEXTURE MAP MANAGER
114
118

FIGURE 1

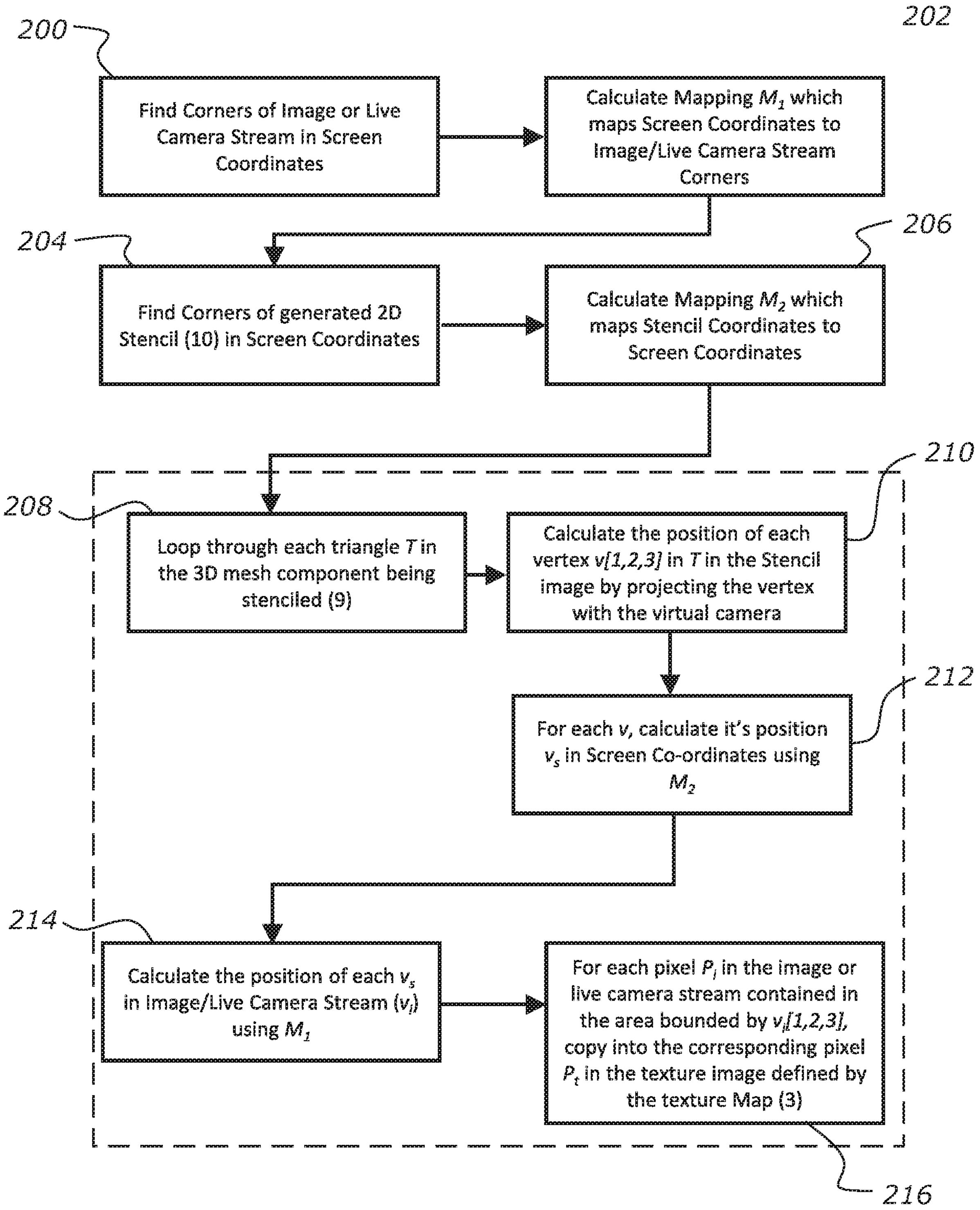
200
202
Find Corners of Image or Live Camera Stream in Screen Coordinates
Calculate Mapping M_1 which maps Screen Coordinates to Image/Live Camera Stream Corners
204
206
Find Corners of generated 2D Stencil (10) in Screen Coordinates
Calculate Mapping M_2 which maps Stencil Coordinates to Screen Coordinates
210
208
Loop through each triangle T in the 3D mesh component being stenciled (9)
Calculate the position of each vertex $v[1,2,3]$ in T in the Stencil image by projecting the vertex with the virtual camera
212
For each v, calculate it's position v_s in Screen Co-ordinates using M_2
214
Calculate the position of each v_s in Image/Live Camera Stream (v_i) using M_1
For each pixel P_i in the image or live camera stream contained in the area bounded by $v_i[1,2,3]$, copy into the corresponding pixel P_t in the texture image defined by the texture Map (3)
216

FIGURE 2

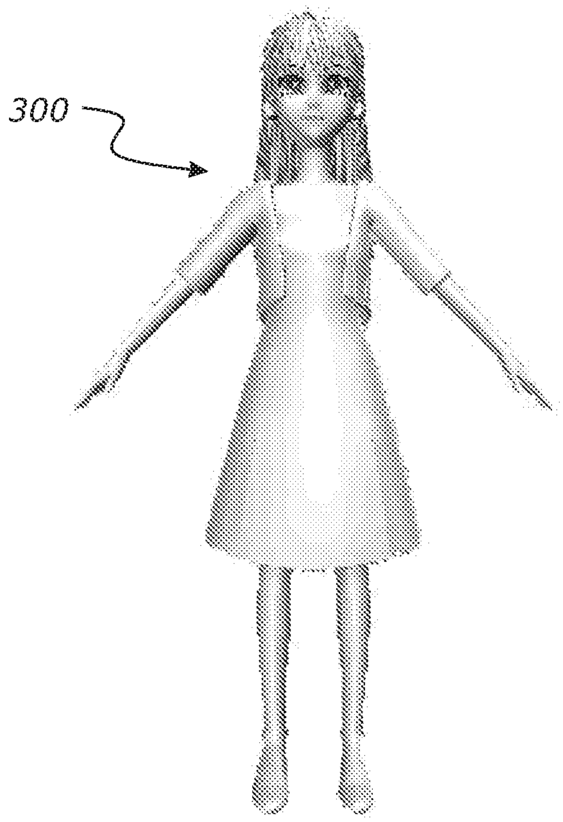

FIGURE 3

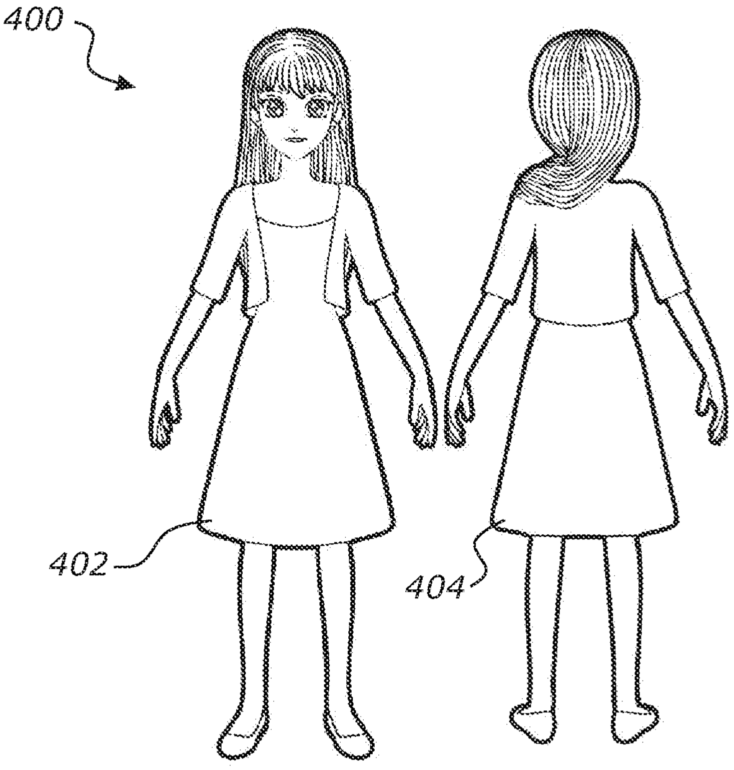

FIGURE 4

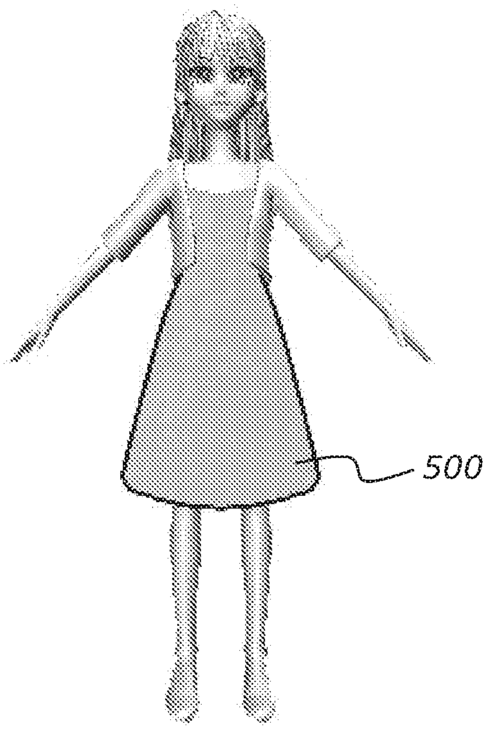

FIGURE 5

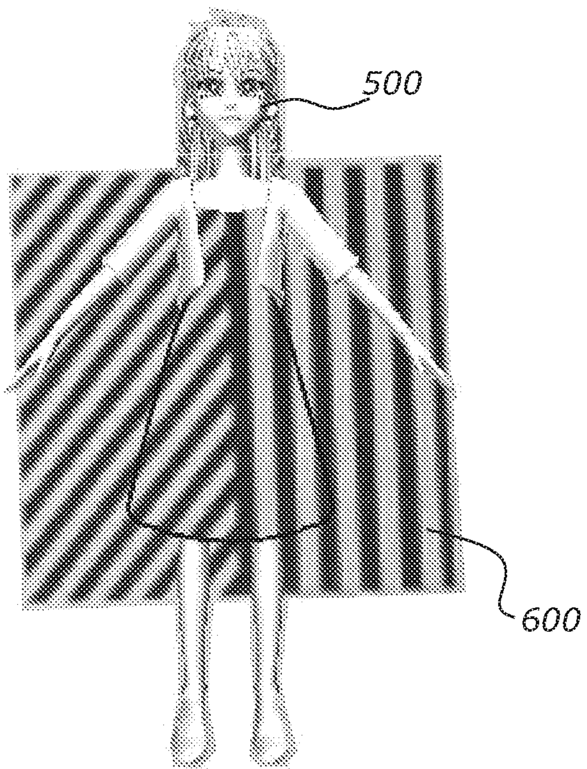

FIGURE 6

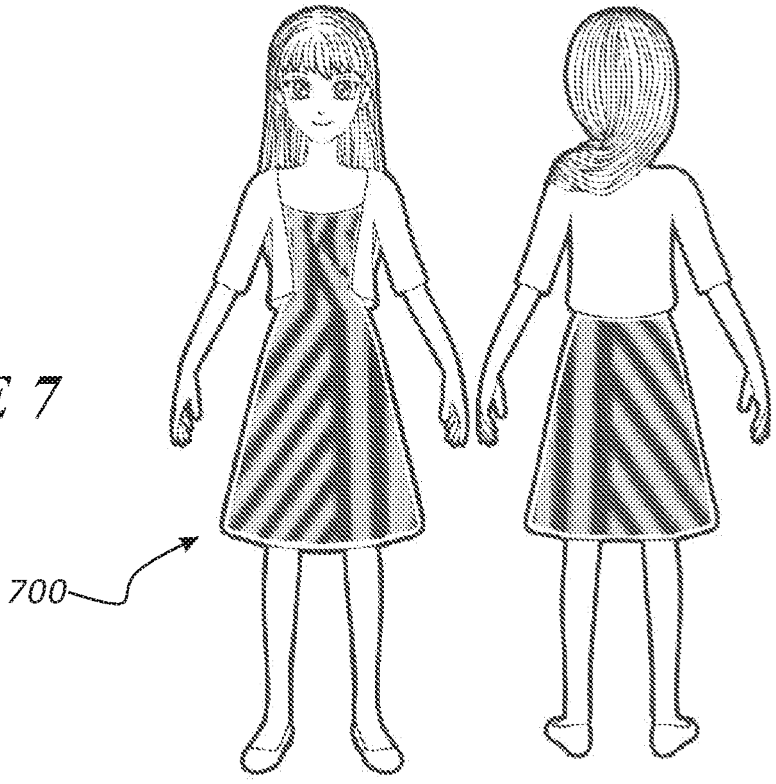

FIGURE 7

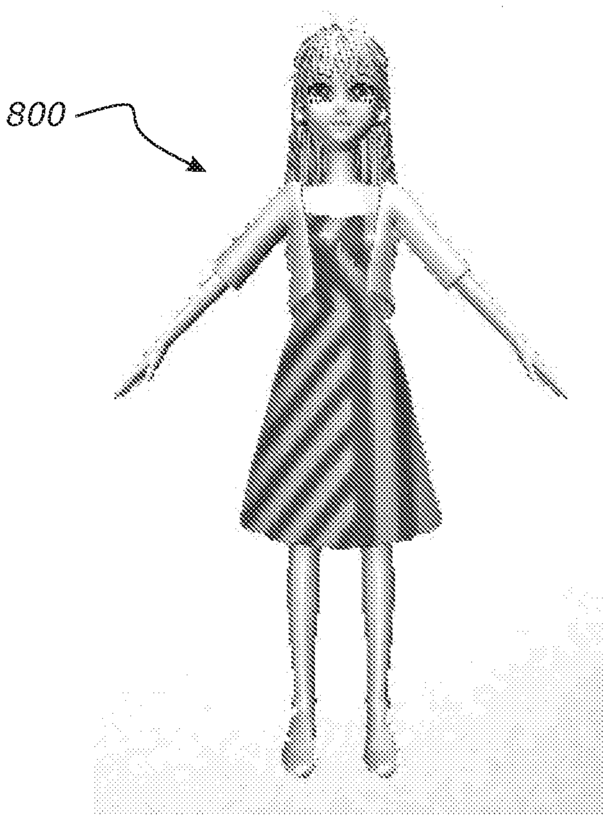

FIGURE 8

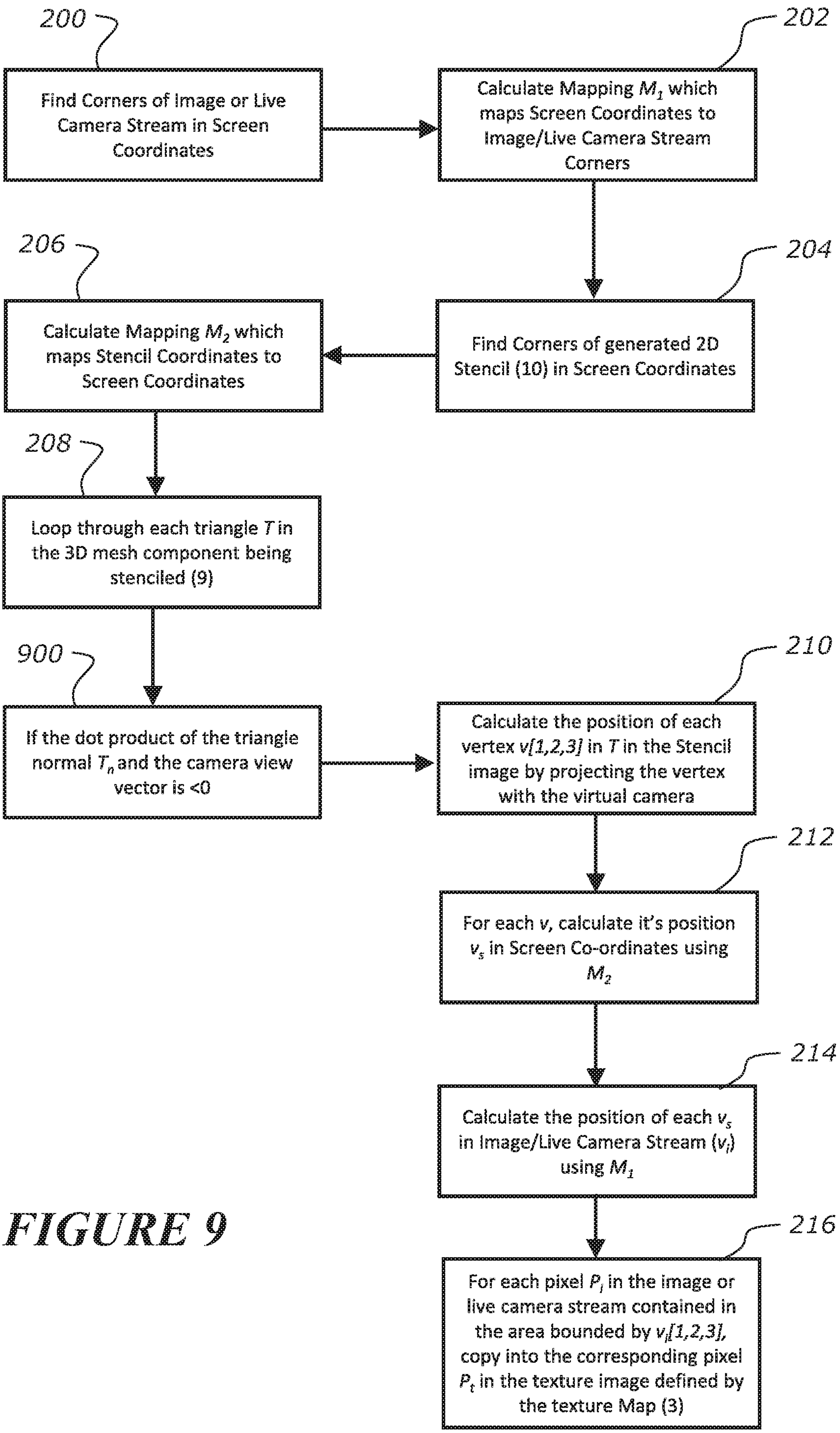
200
Find Corners of Image or Live Camera Stream in Screen Coordinates
202
Calculate Mapping M_1 which maps Screen Coordinates to Image/Live Camera Stream Corners
204
Find Corners of generated 2D Stencil (10) in Screen Coordinates
206
Calculate Mapping M_2 which maps Stencil Coordinates to Screen Coordinates
208
Loop through each triangle *T* in the 3D mesh component being stenciled (9)
900
If the dot product of the triangle normal T_n and the camera view vector is <0
210
Calculate the position of each vertex *v[1,2,3]* in *T* in the Stencil image by projecting the vertex with the virtual camera
212
For each *v*, calculate it's position v_s in Screen Co-ordinates using M_2
214
Calculate the position of each v_s in Image/Live Camera Stream (v_i) using M_1
216
For each pixel P_i in the image or live camera stream contained in the area bounded by $v_i[1,2,3]$, copy into the corresponding pixel P_t in the texture image defined by the texture Map (3)

FIGURE 9

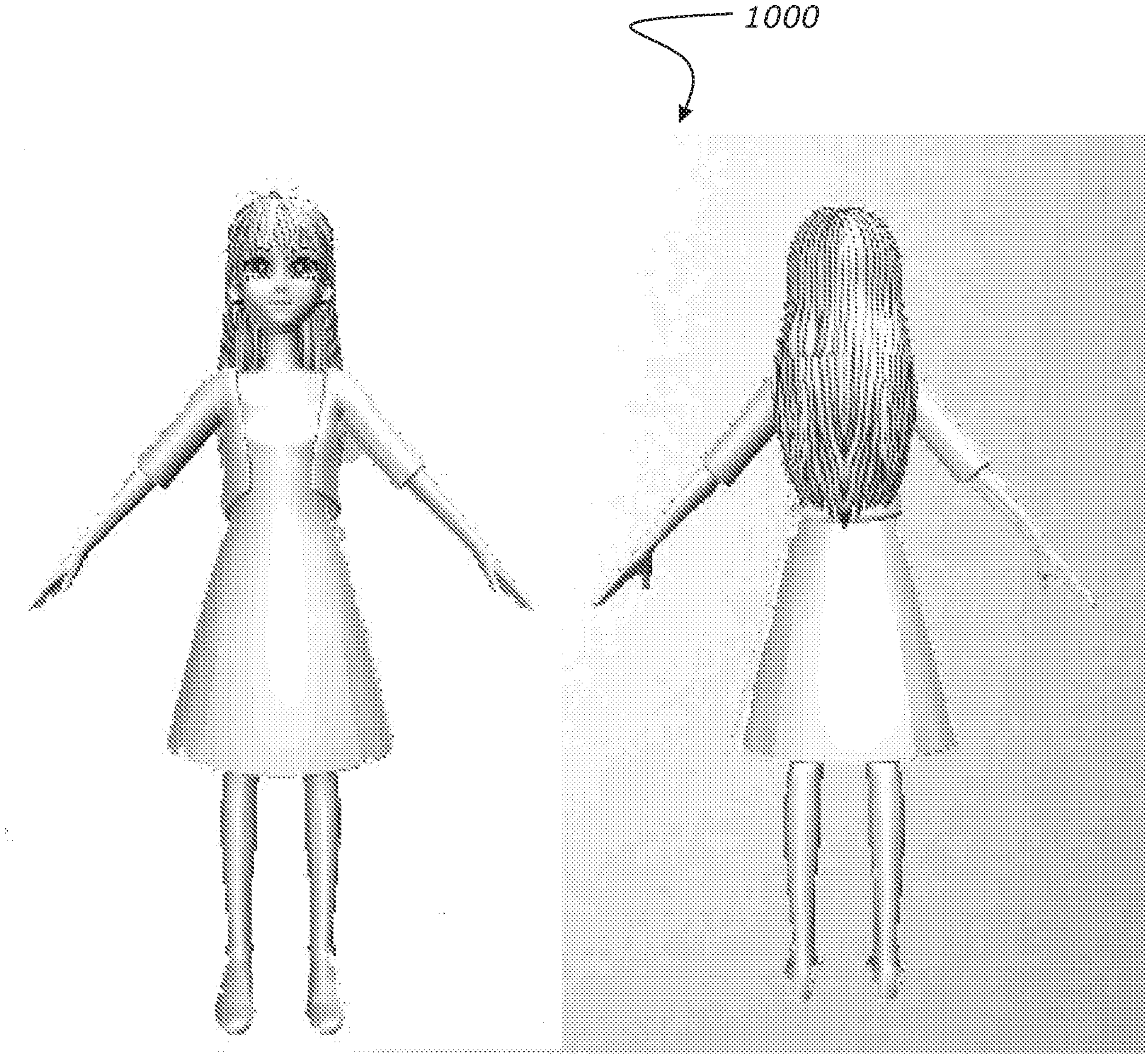

FIGURE 10

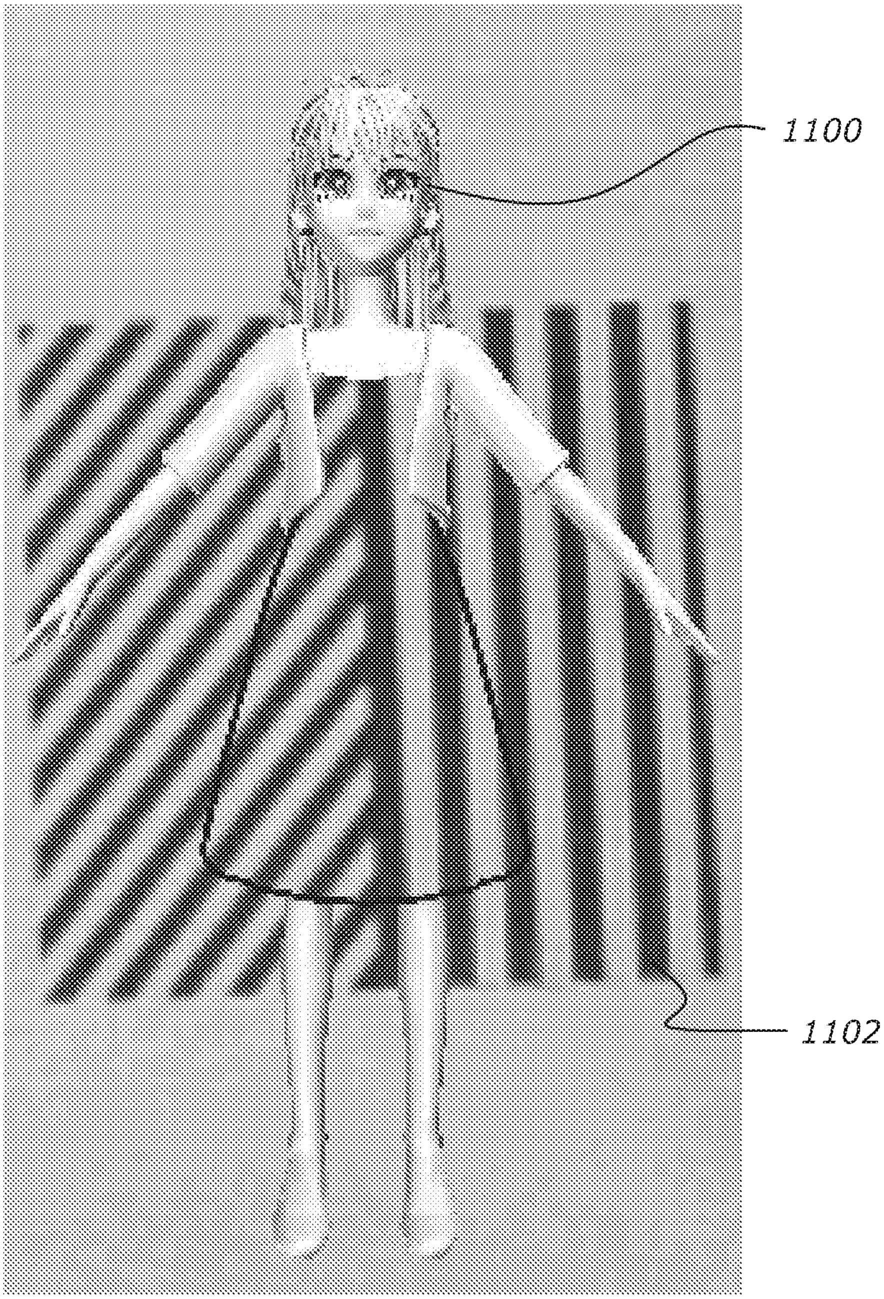

FIGURE 11

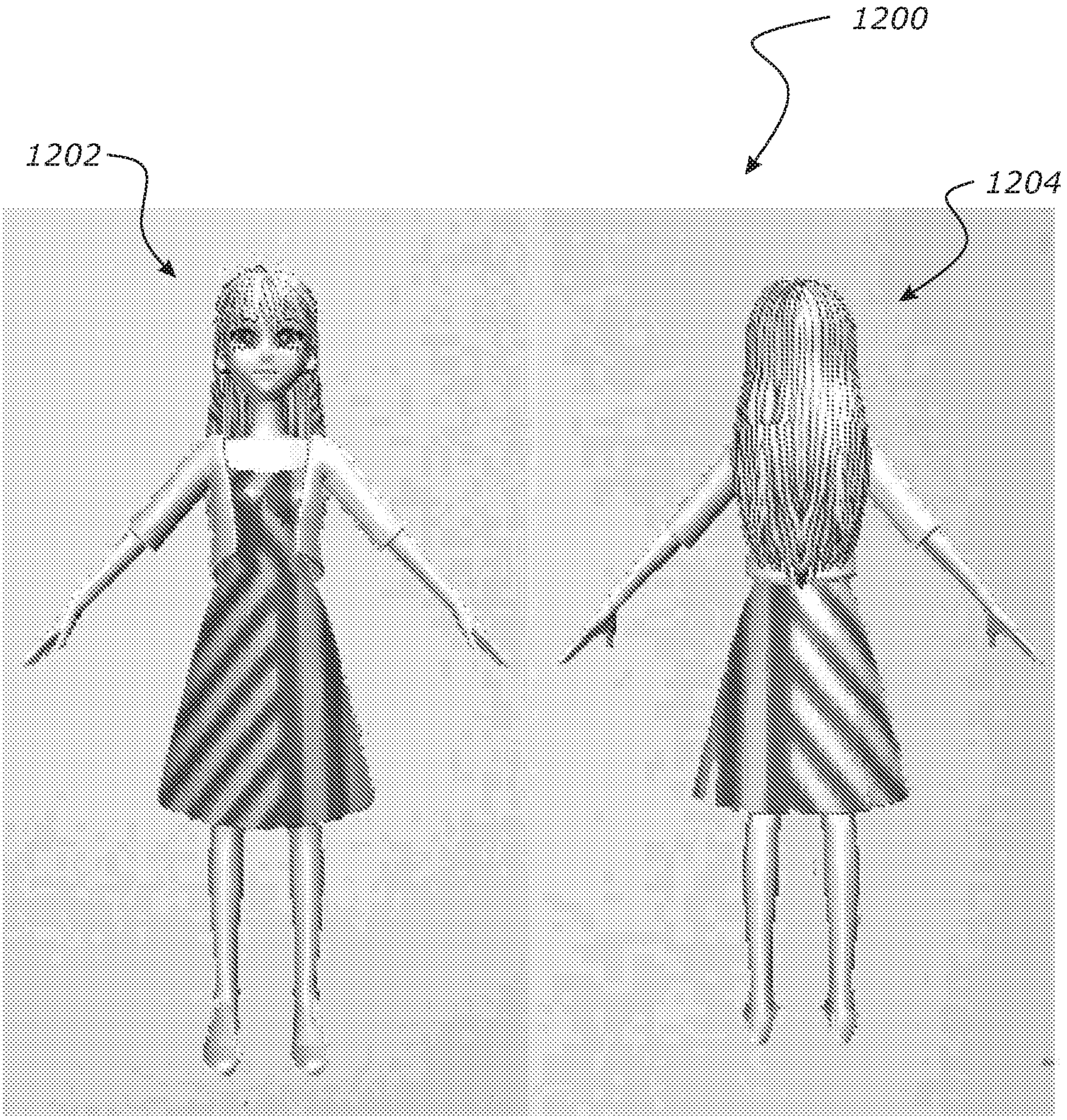

FIGURE 12

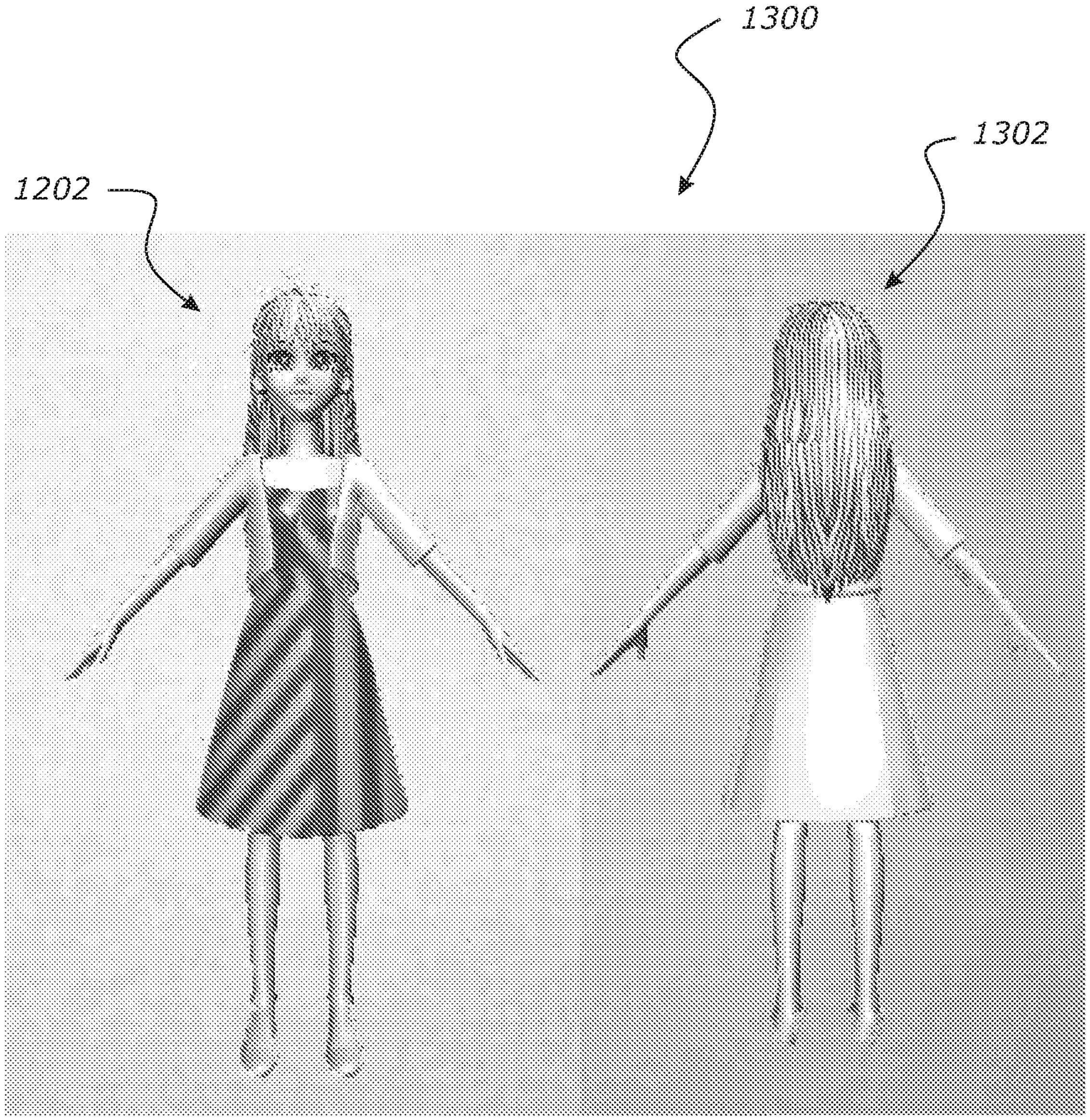

FIGURE 13

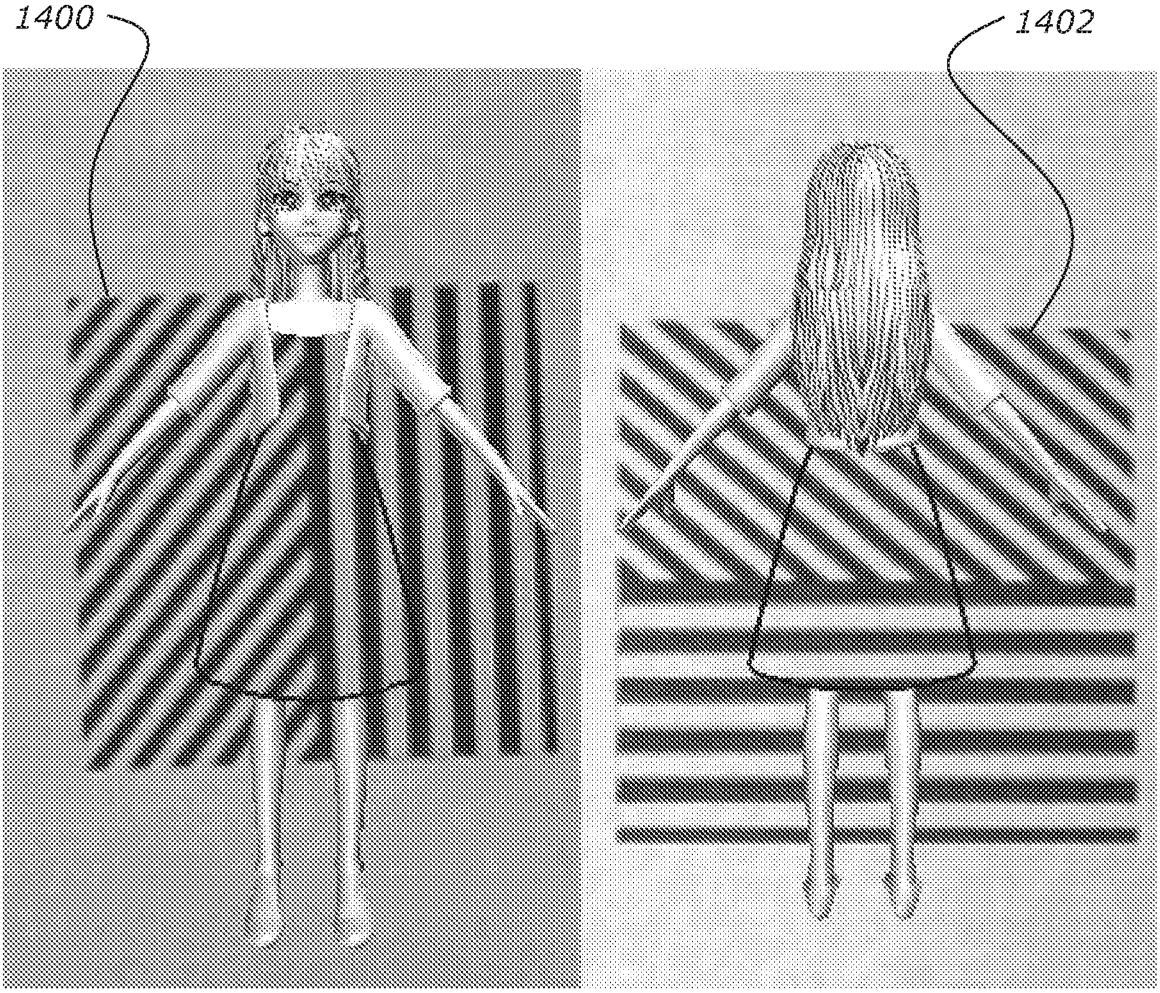

FIGURE 14

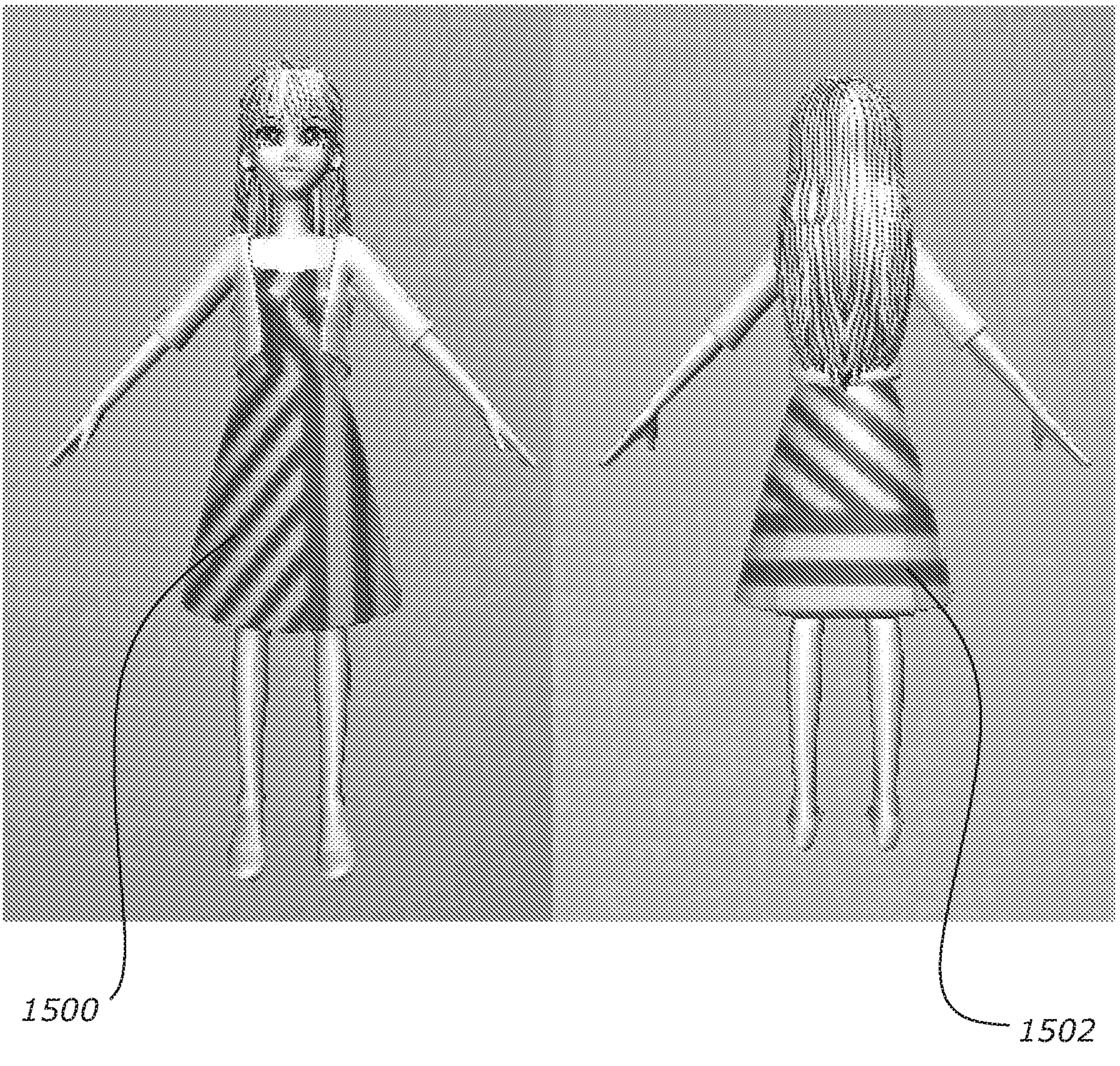

FIGURE 15

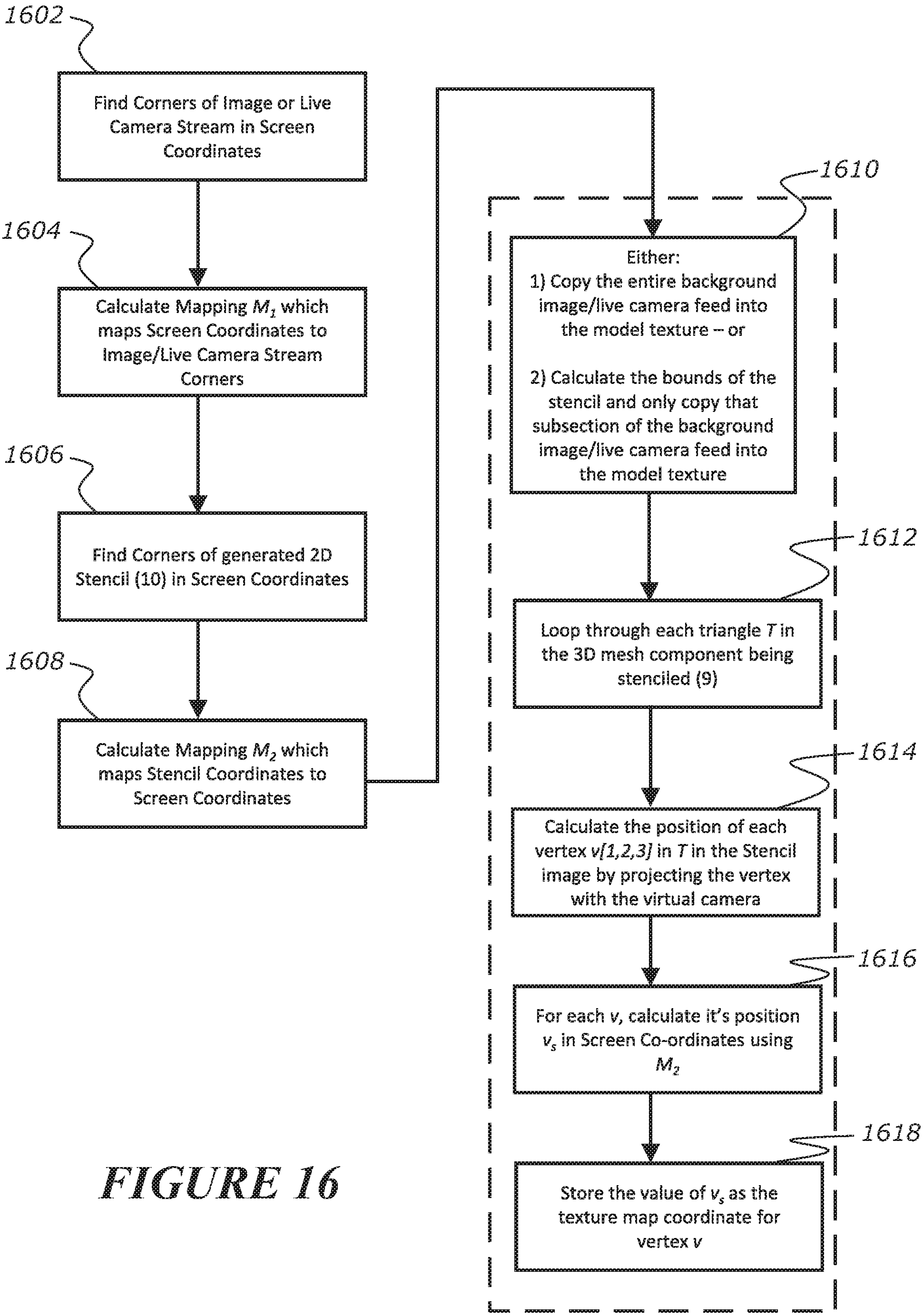
1602
Find Corners of Image or Live Camera Stream in Screen Coordinates
1604
Calculate Mapping M_1 which maps Screen Coordinates to Image/Live Camera Stream Corners
1606
Find Corners of generated 2D Stencil (10) in Screen Coordinates
1608
Calculate Mapping M_2 which maps Stencil Coordinates to Screen Coordinates
1610
Either:
1) Copy the entire background image/live camera feed into the model texture – or
2) Calculate the bounds of the stencil and only copy that subsection of the background image/live camera feed into the model texture
1612
Loop through each triangle *T* in the 3D mesh component being stenciled (9)
1614
Calculate the position of each vertex *v[1,2,3]* in *T* in the Stencil image by projecting the vertex with the virtual camera
1616
For each *v*, calculate it's position v_s in Screen Co-ordinates using M_2
1618
Store the value of v_s as the texture map coordinate for vertex *v*

FIGURE 16

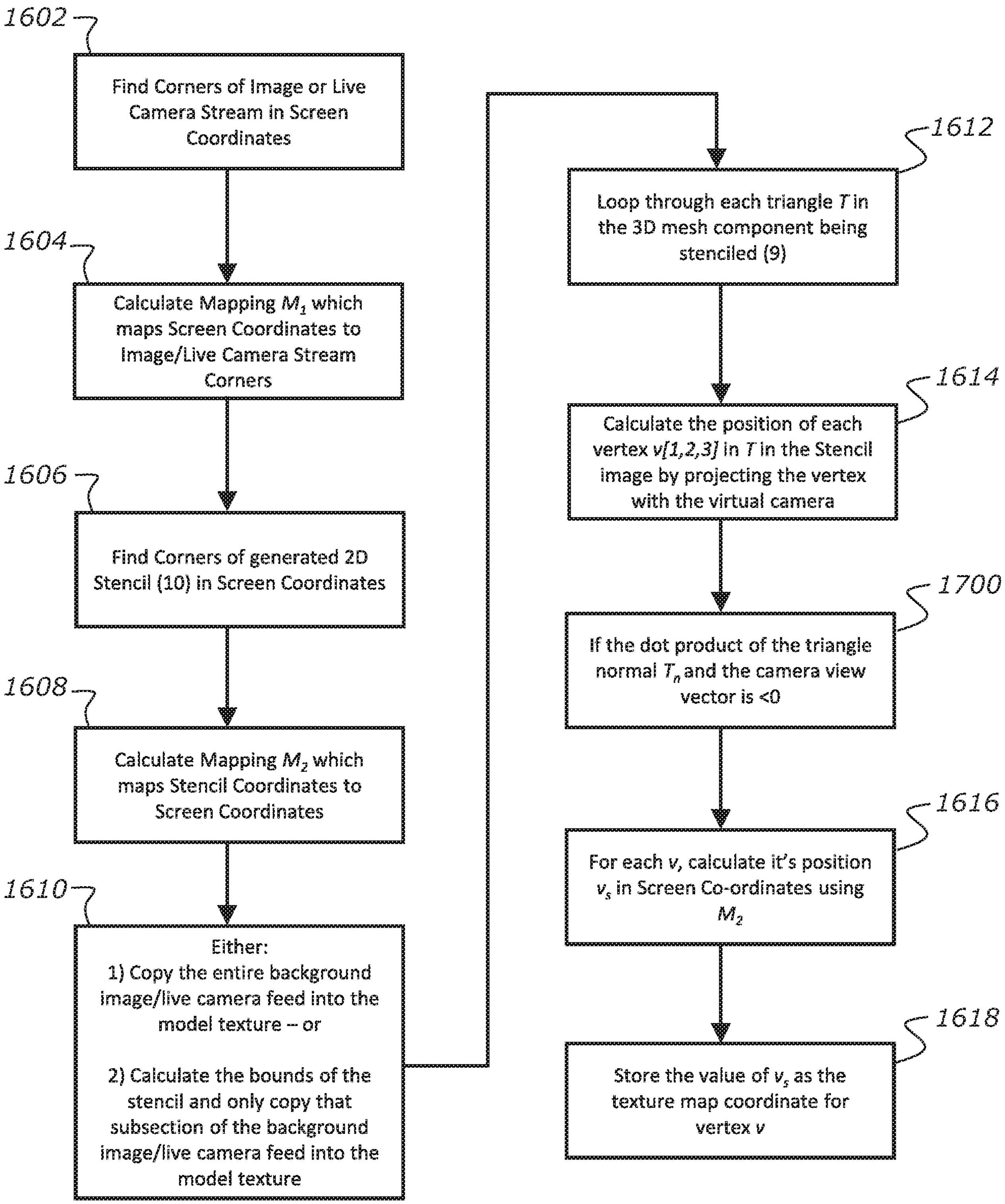
1602
Find Corners of Image or Live Camera Stream in Screen Coordinates
1604
Calculate Mapping M_1 which maps Screen Coordinates to Image/Live Camera Stream Corners
1606
Find Corners of generated 2D Stencil (10) in Screen Coordinates
1608
Calculate Mapping M_2 which maps Stencil Coordinates to Screen Coordinates
1610
Either:
1) Copy the entire background image/live camera feed into the model texture – or
2) Calculate the bounds of the stencil and only copy that subsection of the background image/live camera feed into the model texture
1612
Loop through each triangle *T* in the 3D mesh component being stenciled (9)
1614
Calculate the position of each vertex *v[1,2,3]* in *T* in the Stencil image by projecting the vertex with the virtual camera
1700
If the dot product of the triangle normal T_n and the camera view vector is <0
1616
For each *v*, calculate it's position v_s in Screen Co-ordinates using M_2
1618
Store the value of v_s as the texture map coordinate for vertex *v*

FIGURE 17

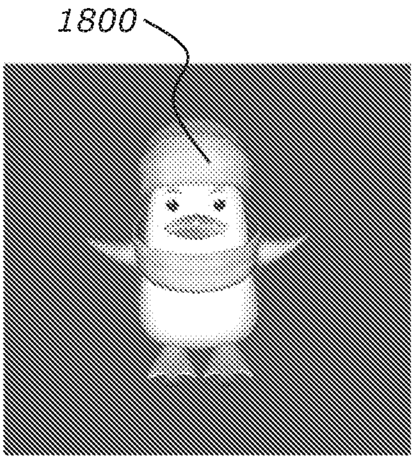

FIGURE 18

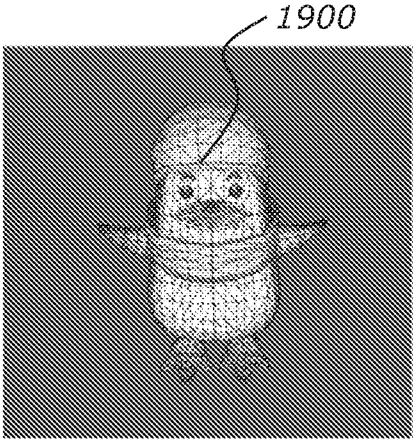

FIGURE 19

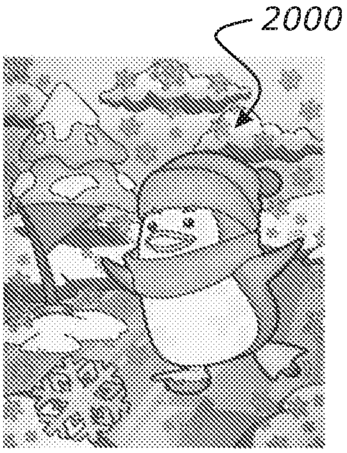

FIGURE 20

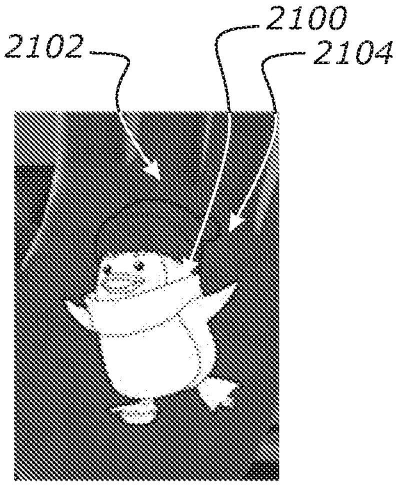

FIGURE 21

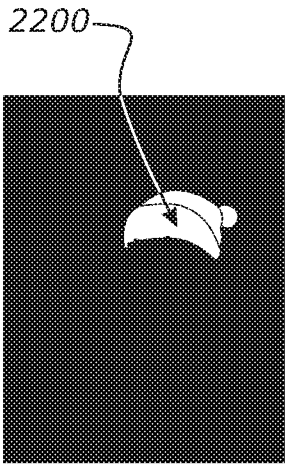

FIGURE 22

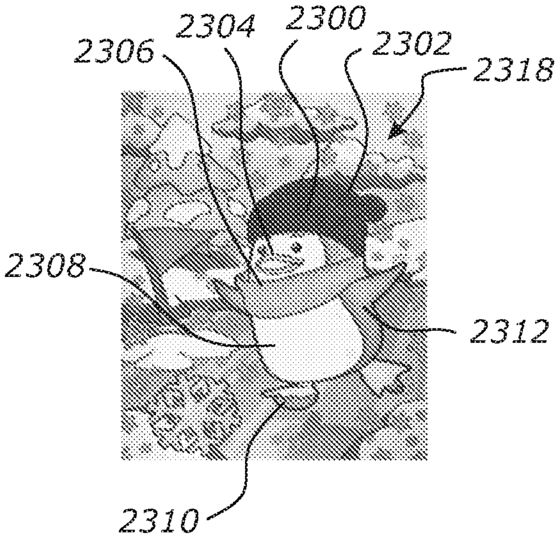

FIGURE 23

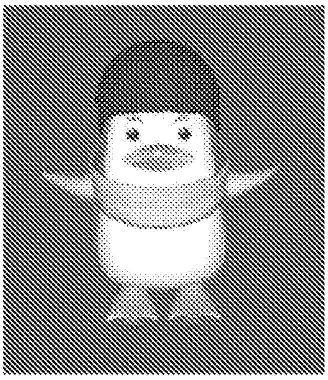

FIGURE 24

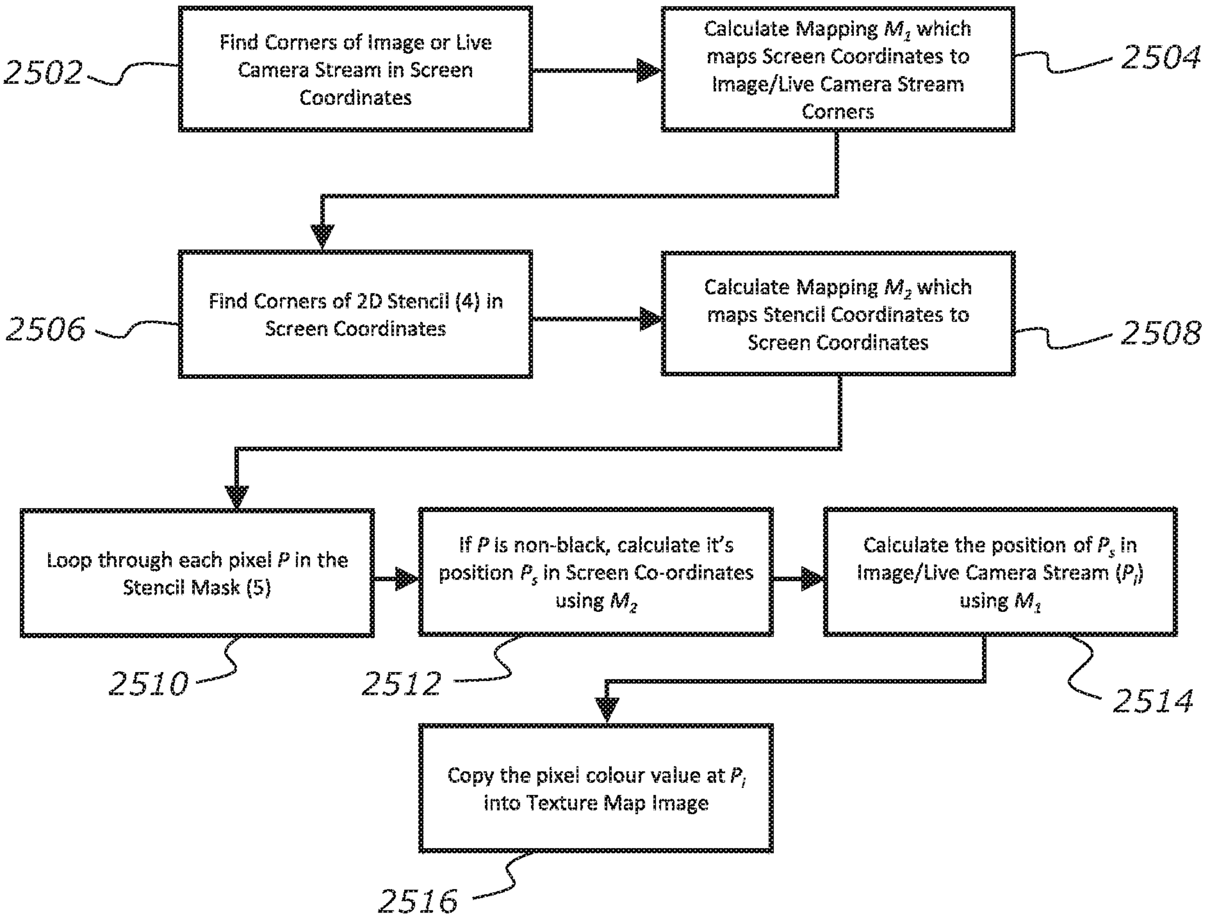
2502
Find Corners of Image or Live Camera Stream in Screen Coordinates
2504
Calculate Mapping M_1 which maps Screen Coordinates to Image/Live Camera Stream Corners
2506
Find Corners of 2D Stencil (4) in Screen Coordinates
2508
Calculate Mapping M_2 which maps Stencil Coordinates to Screen Coordinates
2510
Loop through each pixel P in the Stencil Mask (5)
2512
If P is non-black, calculate it's position P_s in Screen Co-ordinates using M_2
2514
Calculate the position of P_s in Image/Live Camera Stream (P_i) using M_1
2516
Copy the pixel colour value at P_i into Texture Map Image

FIGURE 25

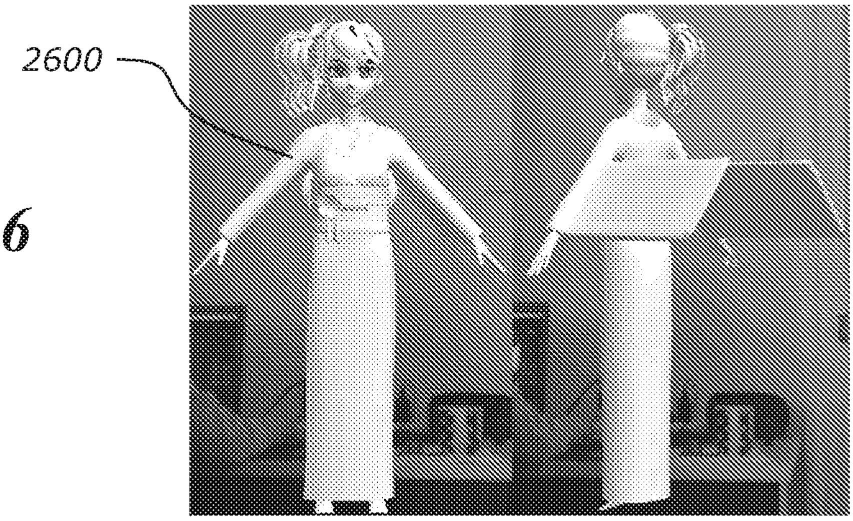

FIGURE 26

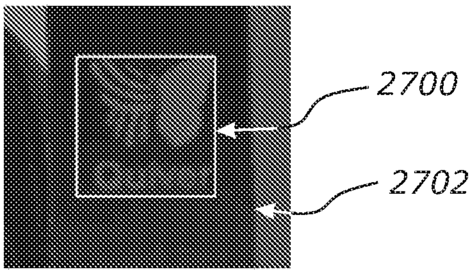

FIGURE 27

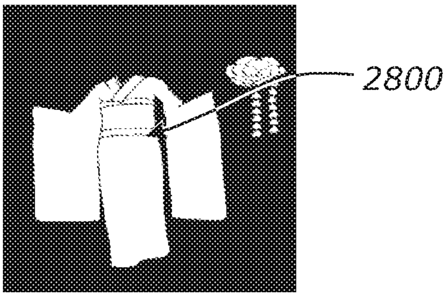

FIGURE 28

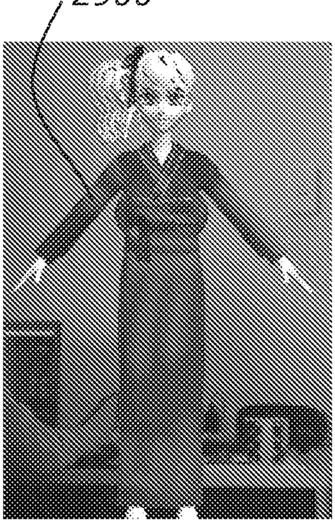

FIGURE 29

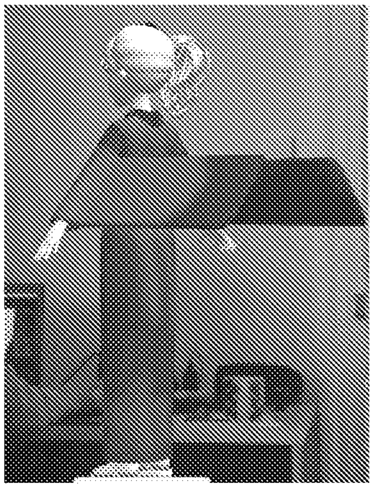

FIGURE 30

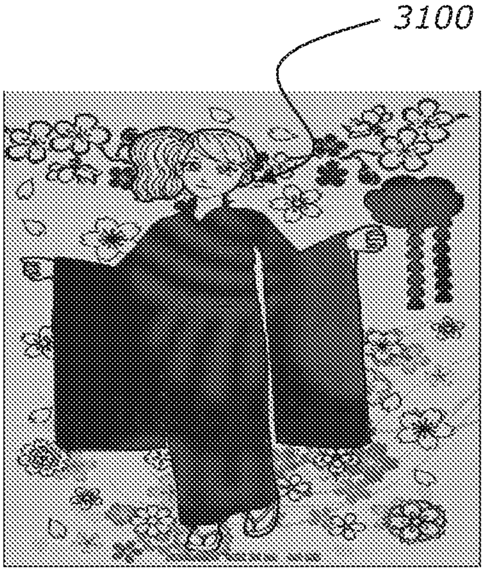

FIGURE 31

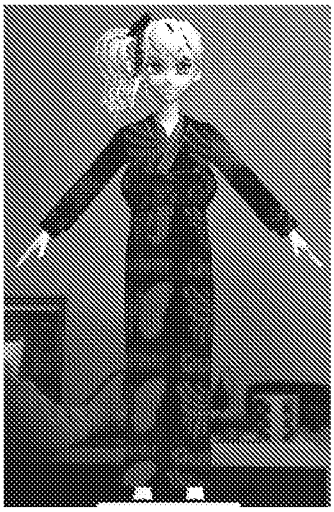

FIGURE 32

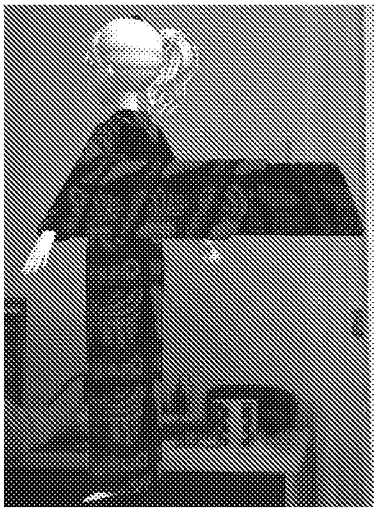

FIGURE 33

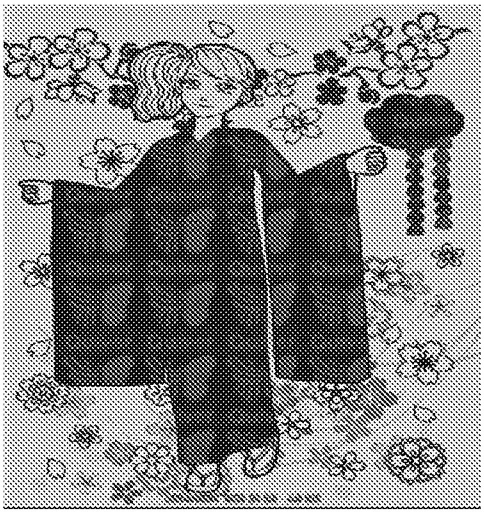

FIGURE 34

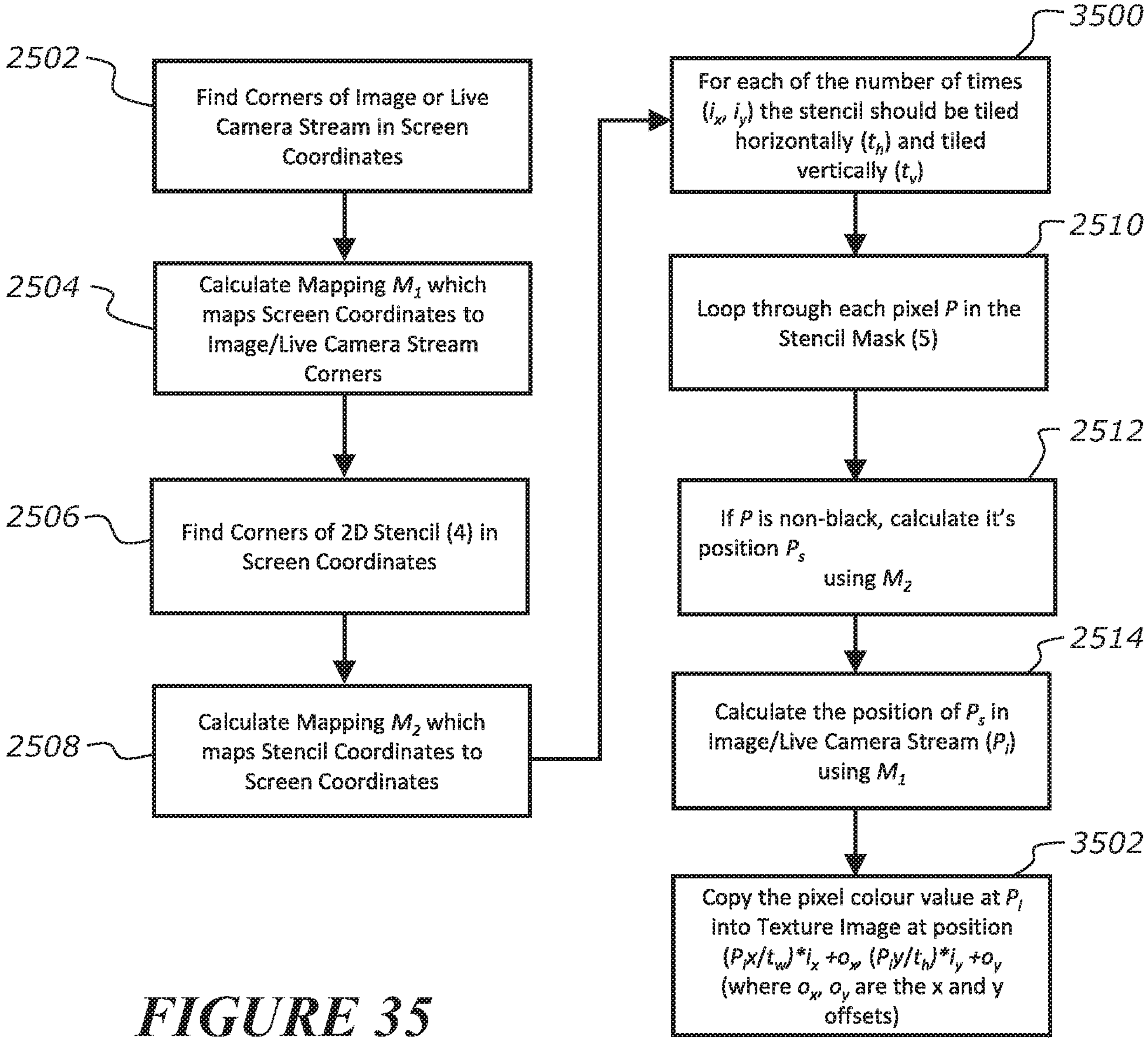
2502
Find Corners of Image or Live Camera Stream in Screen Coordinates
2504
Calculate Mapping M_1 which maps Screen Coordinates to Image/Live Camera Stream Corners
2506
Find Corners of 2D Stencil (4) in Screen Coordinates
2508
Calculate Mapping M_2 which maps Stencil Coordinates to Screen Coordinates
3500
For each of the number of times (i_x, i_y) the stencil should be tiled horizontally (t_h) and tiled vertically (t_v)
2510
Loop through each pixel P in the Stencil Mask (5)
2512
If P is non-black, calculate it's position P_s using M_2
2514
Calculate the position of P_s in Image/Live Camera Stream (P_i) using M_1
3502
Copy the pixel colour value at P_i into Texture Image at position $(P_ix/t_w)*i_x + o_x$, $(P_iy/t_h)*i_y + o_y$ (where o_x, o_y are the x and y offsets)

FIGURE 35

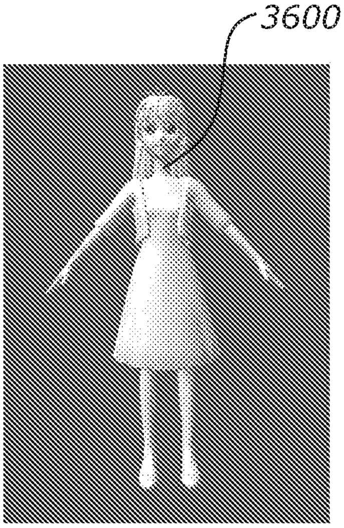

FIGURE 36

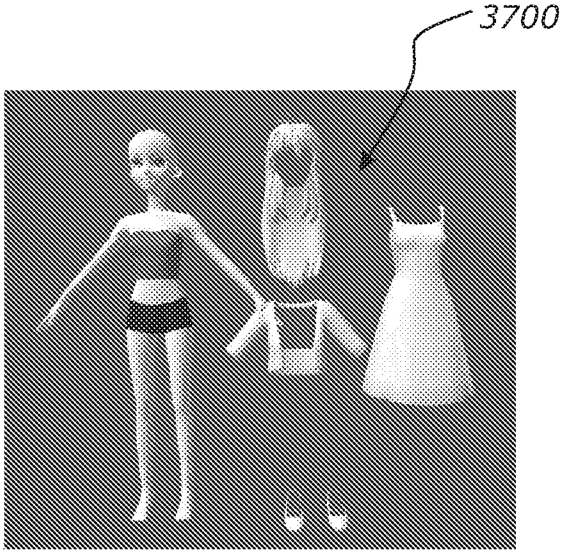

FIGURE 37

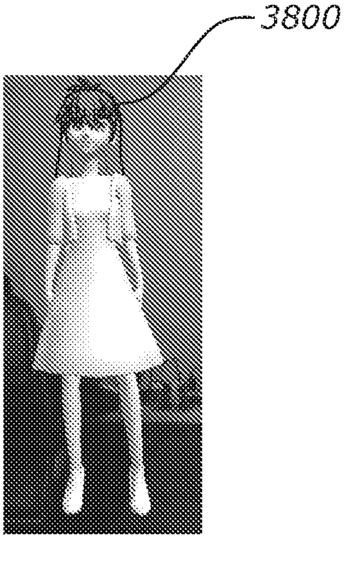

FIGURE 38

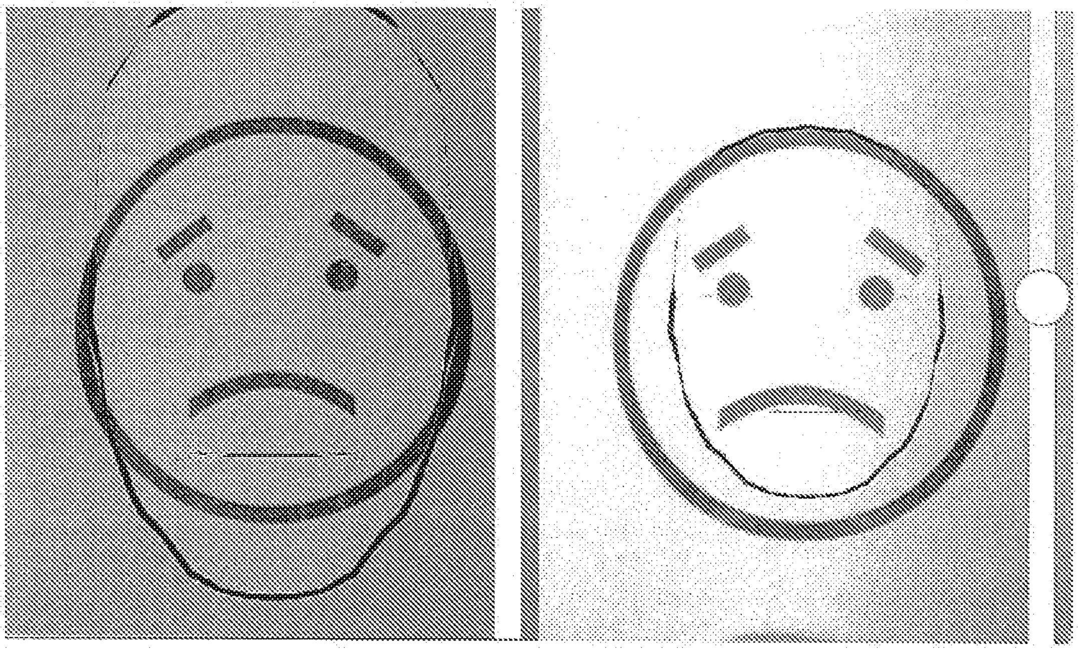

FIGURE 39

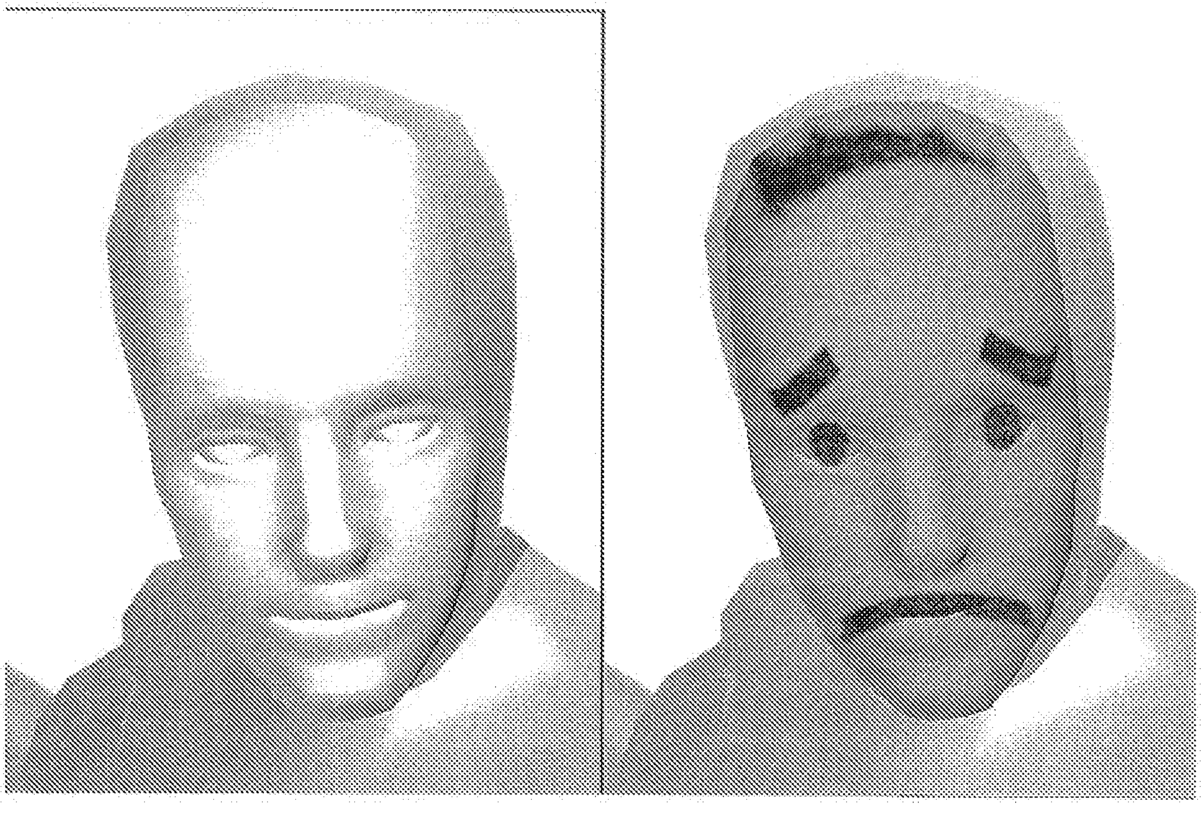

FIGURE 40

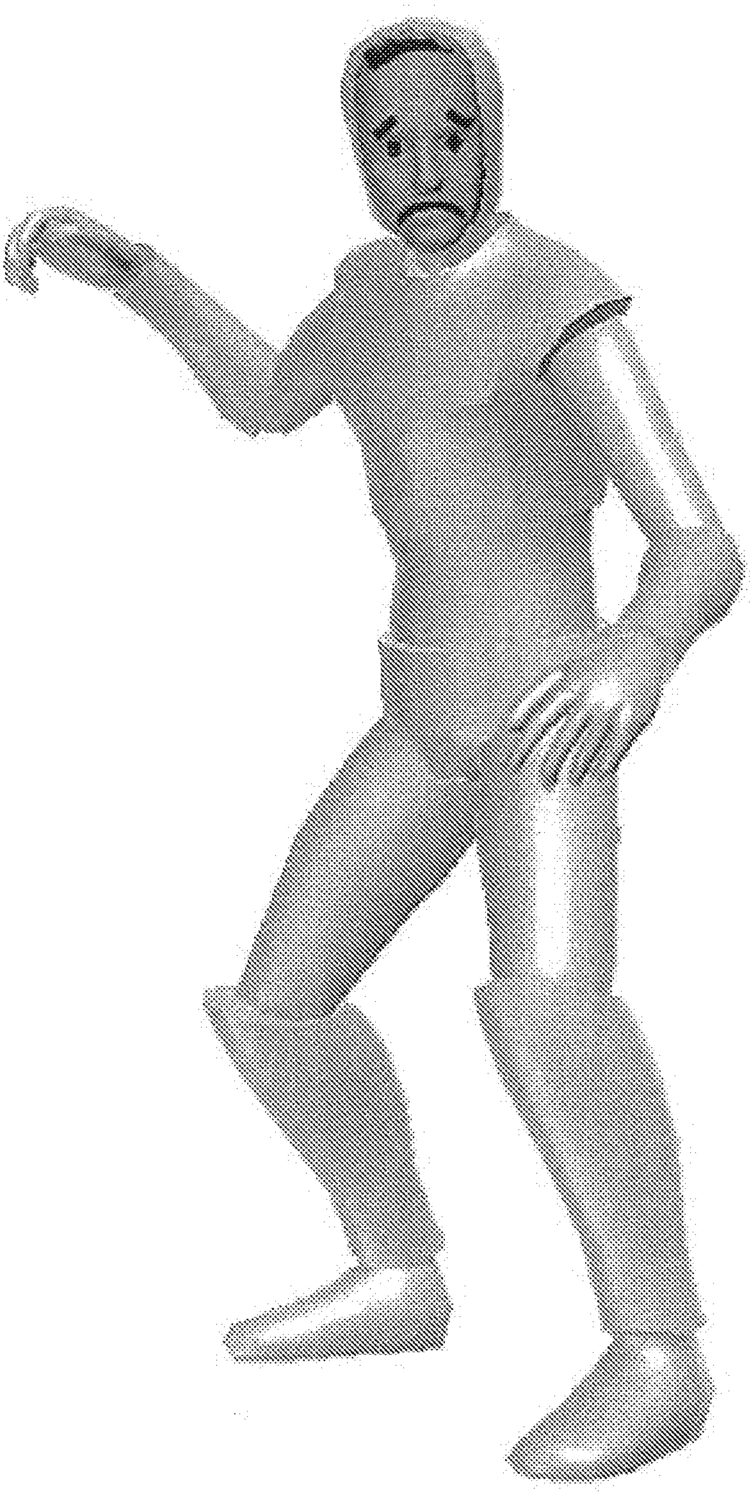

FIGURE 41

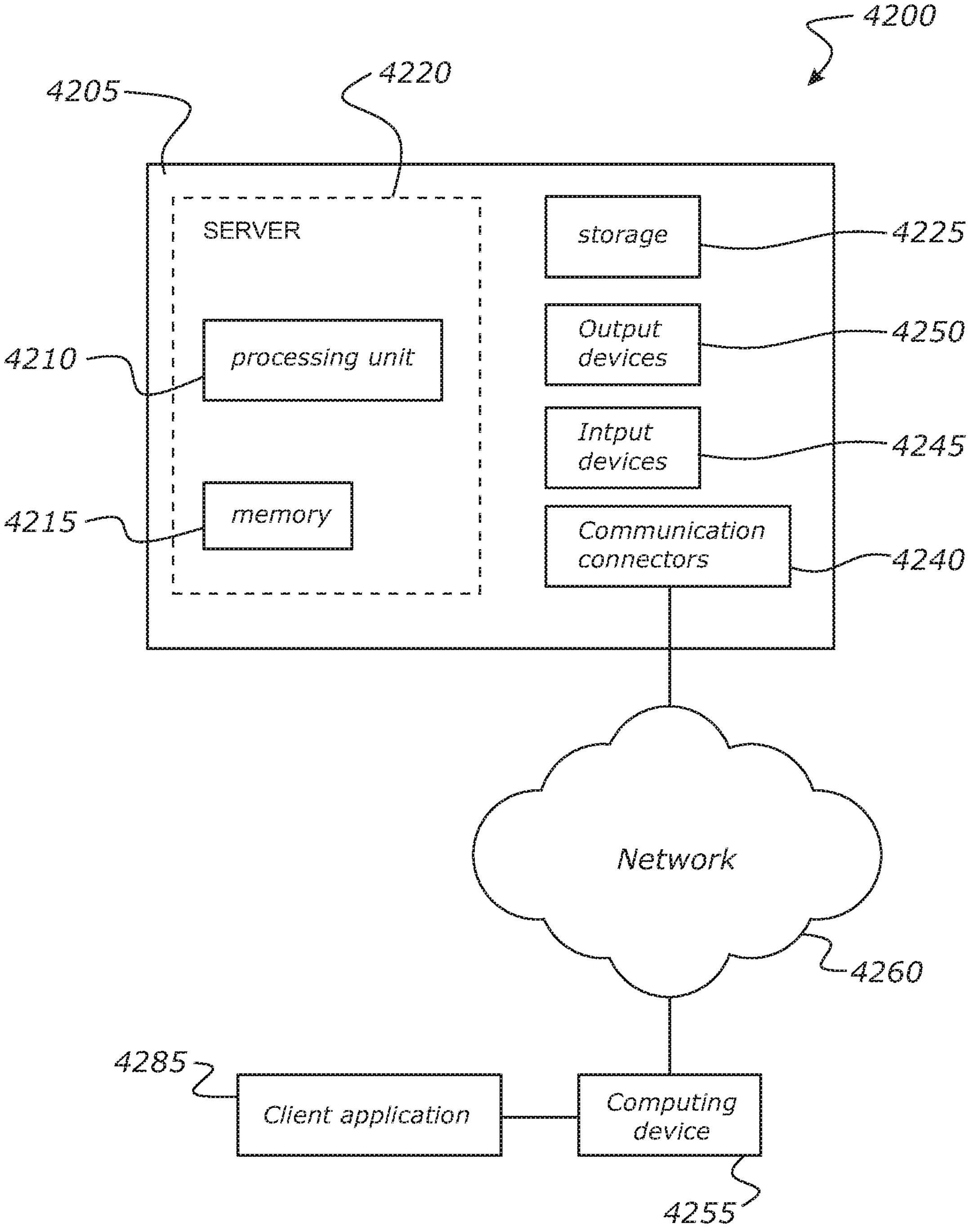
4200
4205
4220
SERVER
processing unit
4210
memory
4215
storage
4225
Output devices
4250
Intput devices
4245
Communication connectors
4240
Network
4260
4285
Client application
Computing device
4255

FIGURE 42

INTERNATIONAL SEARCH REPORT

International application No.
PCT/NZ2018/050013

A. CLASSIFICATION OF SUBJECT MATTER

G06T 15/04 (2011.01) G06T 17/00 (2006.01)

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPOQUE PATENW: IPC/CPC - G06T 15/00; G06T 15/04; Keywords - "texture, map, 3d, model, capture, image, region, stencil" & similar terms.
Google Patents & Scholar: Keywords - "G06T1 5/04, texture, map, model, render, image, stencil, transform, coordinates, region".
Applicant & Inventor Name Search: performed in Espacenet, Google Patents & IP Australia internal databases.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
	Documents are listed in the continuation of Box C	

☒ Further documents are listed in the continuation of Box C ☒ See patent family annex

* Special categories of cited documents:

"A"	document defining the general state of the art which is not considered to be of particular relevance	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"E"	earlier application or patent but published on or after the international filing date	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"O"	document referring to an oral disclosure, use, exhibition or other means	"&"	document member of the same patent family
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 18 May 201 8	Date of mailing of the international search report 18 May 201 8
Name and mailing address of the ISA/AU AUSTRALIAN PATENT OFFICE PO BOX 200, WODEN ACT 2606, AUSTRALIA Email address: pct@ipaustralia.gov.au	**Authorised officer** Ravi McCosker AUSTRALIAN PATENT OFFICE (ISO 9001 Quality Certified Service) Telephone No. +612 6283 2933

INTERNATIONAL SEARCH REPORT

International application No.

PCT/NZ2018/050013

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2005015175 1 A1 (HONG et al.) 14 July 2005 Entire document: especially: Abstract & Fig. 1 & Para 10, 24, 32, 34, 42-59.	1-20
X	US 20150325044 A1 (ADORNABLY INC) 12 November 2015 Entire document: especially: Abstract & Fig. 3A-3E, 4, 10, 11 & Para. 2, 5, 6, 58, 59, 80-210.	1-20
A	Stencil Buffer - Wikipedia [viewed on internet on 11 May 2018] <URL: https://en.wikipedia.org/w/index.php?title=Stencil _buffer&oldid=759 150374 > published on 9 January 2017. Entire document.	1-20
A	US 5864342 A (KAJIYA et al.) 26 January 1999 Entire document.	1-20
A	US 652573 1 B1 (SUITS et al.) 25 February 2003 Entire document.	1-20

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/NZ2018/050013

This Annex lists known patent family members relating to the patent documents cited in the above-mentioned international search report. The Australian Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

Patent Document/s Cited in Search Report		**Patent Family Member/s**	
Publication Number	**Publication Date**	**Publication Number**	**Publication Date**
US 20050 15175 1 Al	14 July 2005	US 2005 15175 1 Al	14 Jul 2005
		US 752883 1 B2	05 May 2009
		GB 2406252 A	23 Mar 2005
		GB 2406252 B	02 Apr 2008
US 20150325044 Al	12 November 2015	US 201 5325044 A 1	12 Nov 2015
US 5864342 A	26 January 1999	US 5864342 A	26 Jan 1999
		AU 6766096 A	05 Mar 1997
		CA 2229027 A 1	20 Feb 1997
		EP 0850462 A2	0 1 Jul 1998
		EP 0850462 Bl	04 Oct 2006
		EP 14341 7 1 A2	30 Jun 2004
		EP 14341 72 A2	30 Jun 2004
		JP 2004046886 A	12 Feb 2004
		JP 3968063 B2	29 Aug 2007
		JP 2004102998 A	02 Apr 2004
		JP 4521 167 B2	11 Aug 2010
		JP H I 15 11277 A	28 Sep 1999
		JP 4540753 B2	08 Sep 2010
		US 5808617 A	15 Sep 1998
		US 5852443 A	22 Dec 1998
		US 5867166 A	02 Feb 1999
		US 5870097 A	09 Feb 1999
		US 5880737 A	09 Mar 1999
		US 5886701 A	23 Mar 1999
		US 5949428 A	07 Sep 1999
		US 5977977 A	02 Nov 1999
		US 5990904 A	23 Nov 1999
		US 59991 89 A	07 Dec 1999
		US 6005582 A	2 1 Dec 1999
		US 6008820 A	28 Dec 1999
		US 6016150 A	18 Jan 2000
		US 6064393 A	16 May 2000
		US 6252608 Bl	26 Jun 2001

Due to data integration issues this family listing may not include 10 digit Australian applications filed since May 2001.

INTERNATIONAL SEARCH REPORT Information on patent family members	International application No. **PCT/NZ2018/050013**

This Annex lists known patent family members relating to the patent documents cited in the above-mentioned international search report. The Australian Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

Patent Document/s Cited in Search Report		Patent Family Member/s	
Publication Number	**Publication Date**	**Publication Number**	**Publication Date**
		US 6292194 Bl	18 Sep 2001
		US 6326964 Bl	04 Dec 2001
		WO 97065 12 A2	20 Feb 1997
US 652573 1 Bl	25 February 2003	US 652573 1 Bl	25 Feb 2003

End of Annex

Due to data integration issues this family listing may not include 10 digit Australian applications filed since May 2001.