>>> Raiders of the pottery GAN
>>> Using 3D Generative adversarial networks for data augmentation in archaeological studies

Name: Celia Cintas[†], Manuel Lucena [‡],
José Manuel Fuertes Garcia[§], Claudio Delrieux [¶]
Date: July 12, 2019

IBM Research

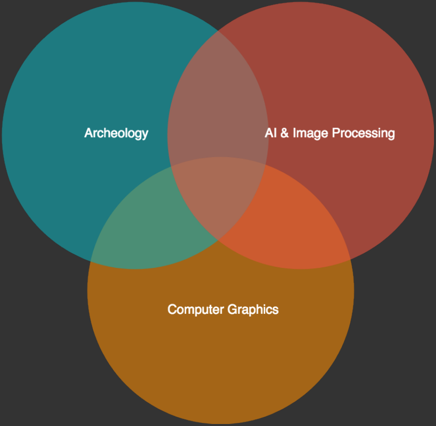
LCI LABORATORIO DE CIENCIAS DE LAS IMAGENES

by mattfrith

[†]celia.cintas@ibm.com / @RTFMCelia - She/Her
[‡]mlucena@ujaen.es
[§]jmf@ujaen.es
[¶]cad@uns.edu.ar

>>> The Team

* José Manuel Fuertes García
* Manuel Lucena López
* Antonio Rueda Ruiz
* Carlos Ogáyar Anguita
* Rafael Segura Sánchez
* Manuel Molinos Molinos
* Arturo Ruiz Rodríguez
* Claudio Delrieux
* Pablo Navarro
* Celia Cintas



IBM Research

LCI LABORATORIO DE CIENCIAS DE LAS IMAGENES

>>> The Problem

Ceramics vessels give a basis for dating the archaeological strata, and provides evidence of local production, trade relations, consumer behavior of the local population, etc. [OTV93, KS03].

Unfortunately, ceramics are fragile and therefore most of the vessels recovered from archaeological sites are broken, so the vast majority of the available material has the form of fragments.

>>> Dataset

The collection corresponds to **Iberian wheel made pottery** from various archaeological sites of the upper valley of the **Guadalquivir River (Spain)**. Contains thousands classified binary images corresponding of the profile views of the pottery. The classification was done by an expert group based on **morphological criteria** (presence or absence of certain parts and ratios between their corresponding sizes) [LMCF+14].

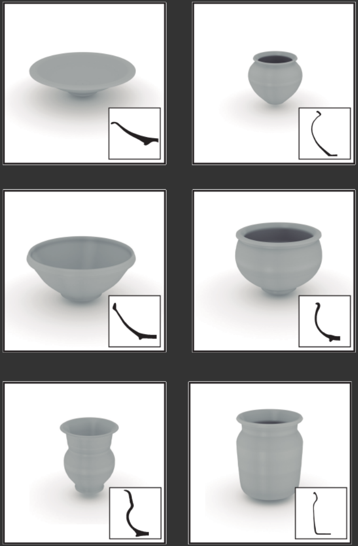
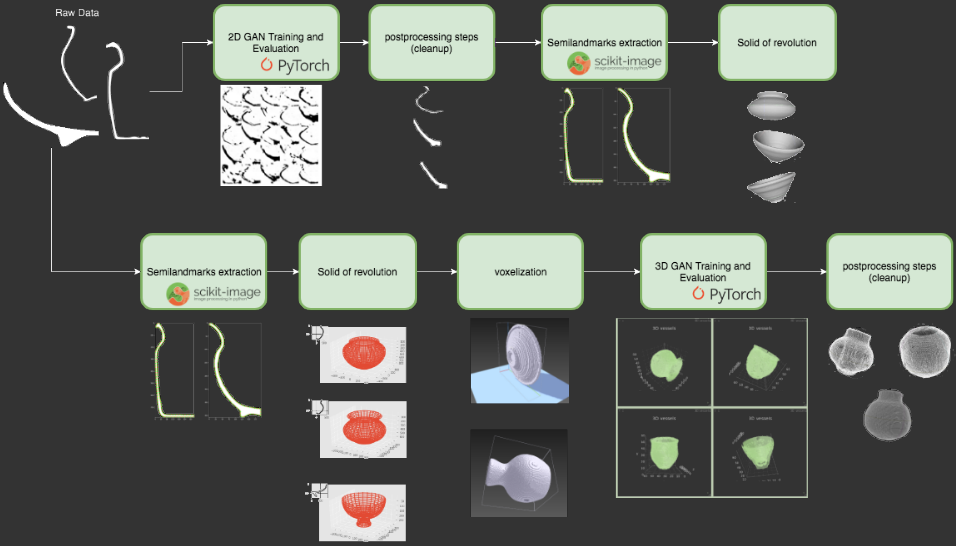
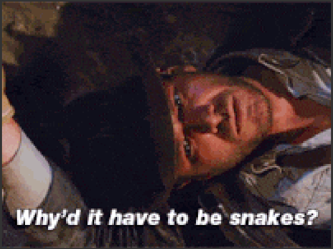


Image from @PabloNavaroooo

>>> Proposed approach



>>> Main Scientific Python libraries used in this work



trimesh PyTorch jupyter

Mind blowing libraries! Don't forget to cite them on your interdisciplinary papers!

>>> What is adversarial?

$$\min_{\texttt{boulder}} \max_{\texttt{indiana}} V(\texttt{indiana}, \texttt{boulder}) = \texttt{distance between them}$$

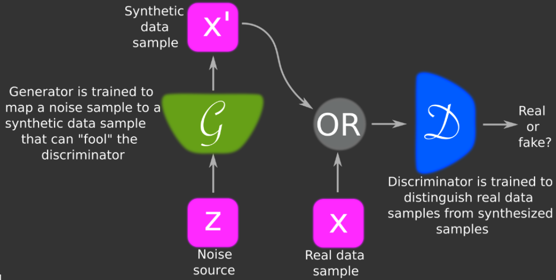
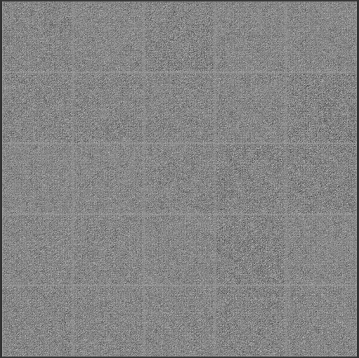


Image from [CWD+17]

>>> Some basic notations

* x will be the data element (🏺).
* $D(x)$ is the discriminator network which outputs the probability that x is real or generated. $D(x)$ should be high when x comes from training data and low when x comes from the generator.
* z will be a latent space vector sampled from a normal distribution. $G(z)$ represents the generator function that which maps the latent vector z to data-space.
* $D(G(z))$ is the probability that the output of the generator G is a real image. D tries to maximize the probability it correctly classifies reals and fakes $(\log D(x))$, and G tries to minimize the probability that D will predict its outputs are fake $(\log(1 - D(G(x)))$

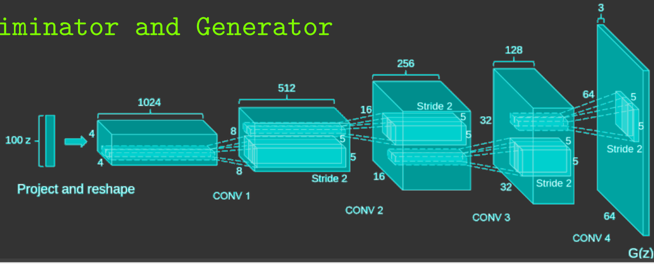
>>> Training a 2D GAN (Cont.)



```
z_ = torch.rand((batch_size, z_dim))
x_, z_ = Variable(x_), Variable(z_)
# update D network
D_real = D(x_)
D_real_loss = BCE_loss(D_real,
                       y_real_)
G_ = G(z_)
D_fake = D(G_)
D_fake_loss = BCE_loss(D_fake,
                       y_fake_)
D_loss = D_real_loss + D_fake_loss
D_loss.backward()
D_optimizer.step()

# update G network
G_ = G(z_)
D_fake = D(G_)
G_loss = BCE_loss(D_fake, y_real_)
G_loss.backward()
G_optimizer.step()
```

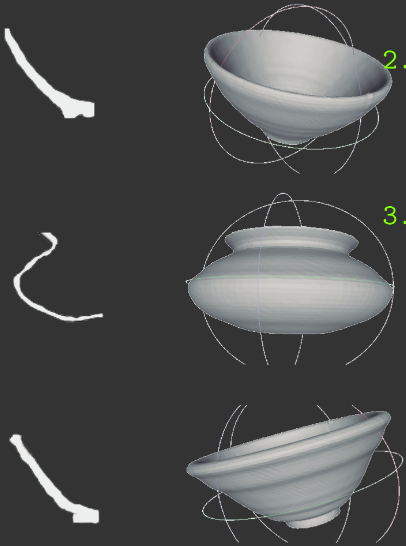
>>> Discriminator and Generator



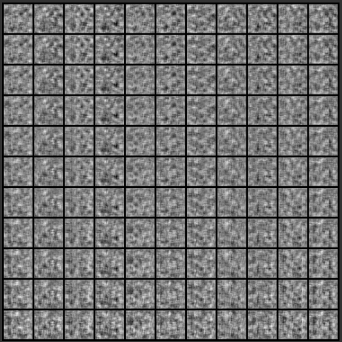

```
nn.Sequential(
        nn.ConvTranspose2d(nz, ngf * XXXXX),
        nn.BatchNorm2d(ngf * XXX),
        nn.ReLU(True),
        nn.ConvTranspose2d(ngf * 8, ngf * XXXXX),
        nn.BatchNorm2d(ngf * XXXX),
        nn.ReLU(True),
        ..... x2
        nn.ConvTranspose2d( ngf, nc, XXXXX),
        nn.Tanh())
```

Figure from DCGAN paper [RMC15]

>>> Output and Evaluation 2D GANs



1. Train CGAN to generate samples from specific classes.
2. We passed generated samples through our classifier from [CLF+19]. The accuracy for our generated samples (1k) was 0.89.
3. PCA overlap for real and fake samples.



>>> Experiments around 3DGAN

The idea behind using 3DGANs is work on volumetric space directly. Several archaeological databases are digitalized with 3D sensors. Most of the fragment information will be in this format.

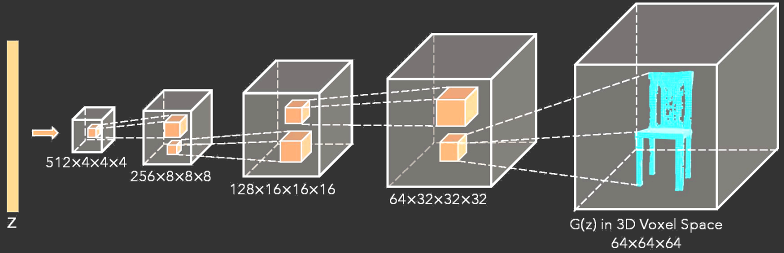
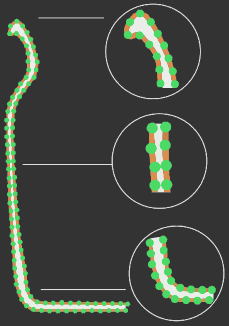


Figure: The generator of 3D Generative Adversarial Networks (3D-GAN) from [WZX+16].

>>> Semi-landmark Extraction

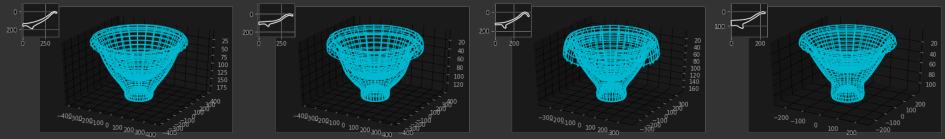


1. The contour extraction was performed using a marching squares algorithm from Scikit-Image. [vdWSN+14, LC87]
2. After the contour is obtained, 200 Semi-landmarks with equidistant position are extracted from the curve.

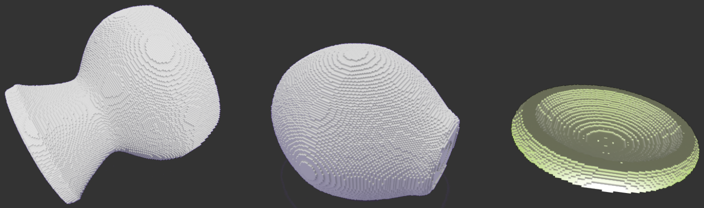
```python
from skimage.io import ImageCollection
from skimage.measure import find_contours

collection = ImageCollection('data/*.png')
for index, img in enumerate(collection):
    contour = find_contours(img,
                            0.8)[0]
    semilands = interpolate(200,
                            contour[:, 0],
                            contour[:, 1])
```

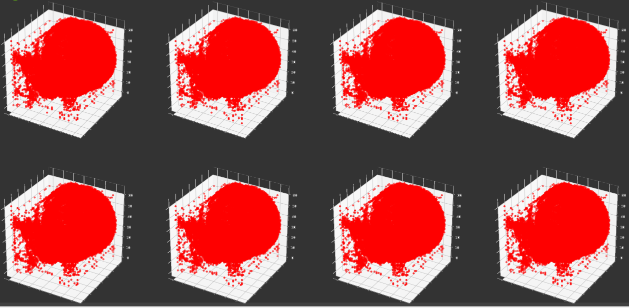
>>> Solid Revolution & Voxelization



```
mesh = trimesh.load(path_to_stl)
mesh.apply_transform(scale_matrix(0.1))
voxels_save = VoxelMesh(mesh, 0.5)
volume = voxels_save.matrix
volume[np.nonzero(volume)] = 1.0
```



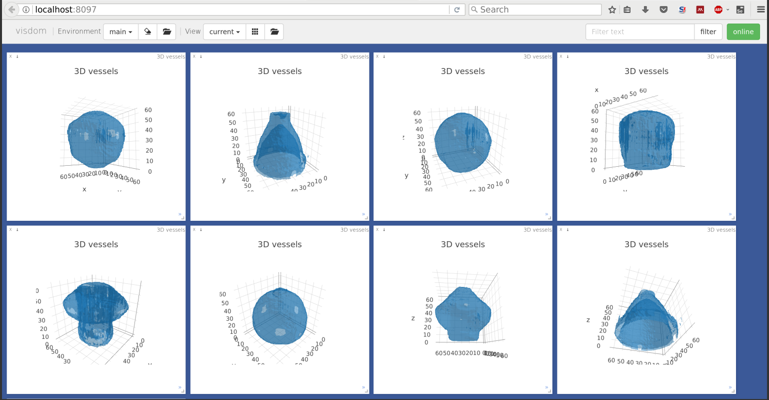
>>> Training the 3D GAN



```
torch.nn.ConvTranspose3d(z_latent_space, cube_len*8, XXXX),
torch.nn.BatchNorm3d(cube_len*8),
torch.nn.ReLU()
torch.nn.ConvTranspose3d(cube_len*8, cube_len*4,XXXXX),
torch.nn.BatchNorm3d(cube_len*4),
torch.nn.ReLU()
torch.nn.ConvTranspose3d(cube_len, 1, XXXX),
torch.nn.Sigmoid()
```

>>> Visualization of output from trained 3DGAN

```
Z = torch.randn(BATCH_SIZE, Z_LATENT_SPACE)
fake = G(Z)
samples = fake.cpu().data[:4].squeeze().numpy()
for sample in samples:
    v, f = sk.marching_cubes_classic(sample, level=threshold)
    vis.mesh(X=v, Y=f)
```



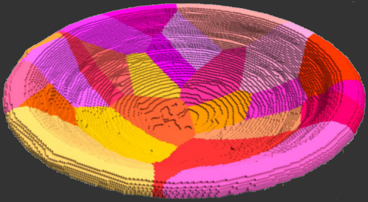
>>> Future work

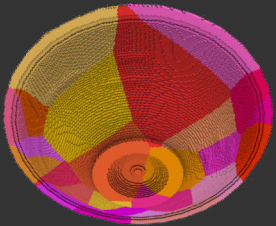
Compare 2DGAN + solid revolution > 3DGAN + remeshing Which methods generates more accurate iberian vessels.

VAE-3DGAN Can we register 3D pottery/fragments from single 2D images?.

Fragment classification Till which size the network is able to localize a fragment?

Add texture information How can we integrate texture information to the geometrical data that we have?





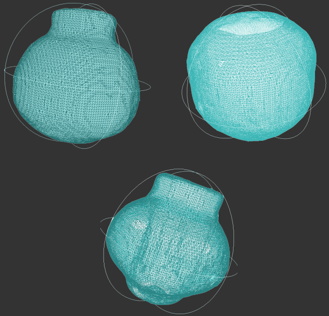
Images from the Computer Graphics team

\>>> Resources

GAN Hacks https://github.com/soumith/ganhacks

3D GANs http://3dgan.csail.mit.edu/

Iberian Vessels Repo https://github.com/celiacintas/vasijas



@RTFMCelia

>>> References I

- Celia Cintas, Manuel Lucena, José Manuel Fuertes, Claudio Delrieux, Pablo Navarro, Rolando González-José, and Manuel Molinos, *Automatic feature extraction and classification of iberian ceramics based on deep convolutional networks*, Journal of Cultural Heritage (2019).

- Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath, *Generative adversarial networks: An overview*, CoRR abs/1710.07035 (2017).

- M. Kampel and R. Sablatnig, *An automated pottery archival and reconstruction system*, Journal of Visualization and Computer Animation 14 (2003), no. 3, 111--120.

- William E. Lorensen and Harvey E. Cline, *Marching cubes: A high resolution 3D surface construction algorithm*, ACM SIGGRAPH Computer Graphics (1987).

>>> References II

- M. Lucena, A. L. Martínez-Carrillo, J.M. Fuertes, F. Carrascosa, and A. Ruiz, *Decision support system for classifying archaeological pottery profiles based on mathematical morphology*, Multimedia Tools and its Applications (2014).
- C. Orton, P. Tyers, and A. Vinci, *Pottery in Archaeology*, Cambridge University Press, United Kingdom, 1993.
- Alec Radford, Luke Metz, and Soumith Chintala, *Unsupervised representation learning with deep convolutional generative adversarial networks*, arXiv preprint arXiv:1511.06434 (2015).
- Stéfan van der Walt, Johannes L. Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D. Warner, Neil Yager, Emmanuelle Gouillart, Tony Yu, and the scikit-image contributors, *scikit-image: image processing in Python*, PeerJ 2 (2014), e453.

>>> References III

Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum, *Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling*, Advances in neural information processing systems, 2016, pp. 82--90.