LUCA
SCHOOL
OF
ARTS

lerend netwerk: Luca

Meeting 25/04/2022 Luca Gent

topics

1. Polymath lab (?)
2. BabelAR
3. TrustVR
4. PWO ondersteuning

@Valery zouden we polymath vermelden?
Polymath:

- Kasper & Wouter
- onderzoekslabo en community: brug technologie - creativiteit

Doel: hoe future creatives empoweren in functie van opkomende technologiën

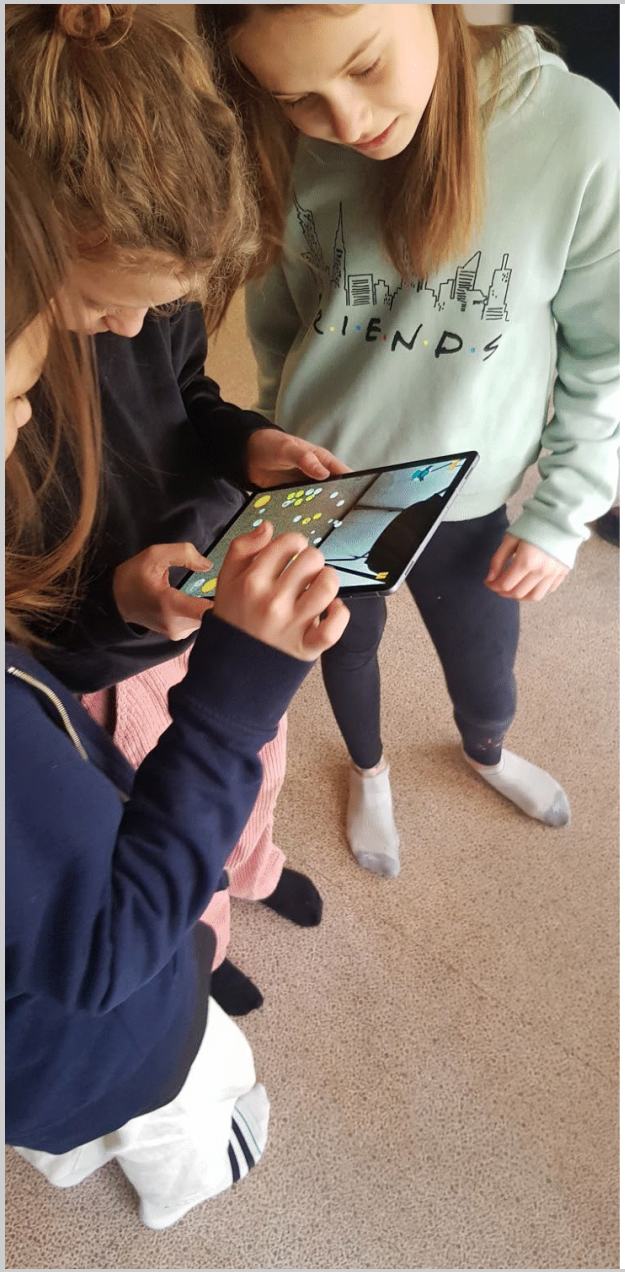

BABELAR

BabelAR

- Erasmus+ project 'virtual language app'
- een meertalige, multi-player Augmented Reality game.
- Henry van de Velde Awards

GOALS GAME

- Create positive multilingual awareness for teachers in primary schools
- Play with multilingualism in classrooms can be fun & augment brain plasticity
- Augment self-confidence pupils about languages



virtulapp

LUCA
SCHOOL
OF
ARTS

SPECS GAME

- 15 languages
- 7 - 12 years (with difficulty levels)
- Collaborative – multiplayer(2-6p)
- More languages possible!
- Android & iOS (Preferred on Tablet)
- Where’s Waldo?! in Augmented Reality
- Text to speech


BABELAR
BELGIAN GAME AWARDS 21
NOMINEE
BEST
NON-ENTERTAINMENT

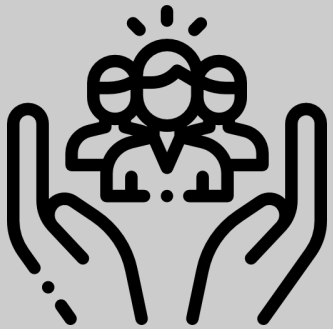
BabelAR film

Rol van Luca

- idee - concept - realisatie
- visual first aanpak:
  - interactieve AR omgeving
  - game-ontwikkeling ism research:
- game: groot bereik
  - meer feedback
  - impact & gekendheid

TrustVR

- samenwerking KULeuven Psychologie en Luca
- hechtingsproblematiek bij kinderen (9-12)
    - invloed van Oxytocine op hechting?



doel

immersieve opstelling :

- VR-omgeving
- data-captatie:
  - EEG, GSR, eye-tracking, questionnaires
- rollenspel: Cyberball

Rol van luca

- ontwikkeling spelomgeving + integratie sensoren
- user story uitschrijven in functie van onderzoek KUL
- samen kijken met KUL voor meerwaarde dmv Virtual Reality
    - game development
    - 3D modelling voor de 'suspension of disbelief'
    - design process

PWO ondersteuning

- domein- + technische experts
- bv Narafi

```
for x in range(2):
  x_tensor = tf.convert_to_tensor(latent_vector, dtype=tf.float32)

  with tf.GradientTape() as t:
      t.watch(x_tensor)
      output = model(x_tensor)

  gradients = tf.gradients(output, x_tensor)

  #with tf.Session() as sess:
  sess.run(tf.global_variables_initializer())
  result_output=np.array(sess.run(gradients)).reshape(18,512)

  latent_vector += result_output
  print(np.mean(model.predict(latent_vector)))
  img = generate_image(latent_vector)
  display(img.resize((512,512),PIL.Image.LANCZOS))

predictions = model.predict(latent_vector).reshape(-1)
print(predictions)

img = generate_image(latent_vector)
display(img.resize((512,512),PIL.Image.LANCZOS))
img.save("generated_images/LB_0001_01.png", "PNG")
```