Project context: Virtual Tribunals

- Collaborative initiative between Stanford Libraries and WSD Handa Center for Human Rights and International Justice
- Technical work part of a larger grant project to provide access to documents from Special Panels for Serious Crimes in East Timor
- Situated in how Digital Library Systems and Services and Stanford Libraries approaches projects and infrastructure
Scope of work

- Provide “search within” a particular SDR object’s pre-existing text (e.g. OCR or transcription), including phrase searching
- Support subset of OCR formats: ALTO, plain text
- Implement parts of Content Search API specifications needed to support use case
content_search: the application

- [https://github.com/sul-dlss/content_search](https://github.com/sul-dlss/content_search)
- Dependencies: Ruby, Rails, Solr, Redis
- Tightly integrated with SDR infrastructure
Indexing process

- Identify new/changed objects (or object for one-off indexing)
- Fetch object structure
- Identify applicable files (e.g. OCR/transcription)
- Read files, transform content, and index to Solr
Text transformation for indexing

- Text and word boundaries extracted from source file
- Each `alto:TextBlock` treated as single value in a multivalued field
- Each `alto:TextLine` delimited by \n in each field
- Concatenate each word with its boundaries using a symbol to generate a Lucene payload:
Metadata and access integration

- Manifests dynamically generated from our delivery systems
- Incorporation of content search services into manifests triggered by structural metadata
- Any changes to objects (addition of new OCR resources) will lead to transparent updates
Example

- homepage
- manifest
  - search service
    - autocomplete service
  - rendering: source PDF and searchable PDF (not from content_search)
  - Canvas: no text annotations or seeAlso (yet)
Search API flow

- Search Solr for each word for terms or phrase and get the matching pages with hit highlighting
- Extract text preceding and following matches to return before and after
- Transform each hit highlight into an Annotation that is on a Canvas URI fragment identified by word boundaries
- Create the hits and response
Autocomplete API flow

- Search Solr (with custom suggester) for requested string
- Remove duplicate matches
- Sort by occurrence (“weight”) and then by length
- Gather the top 5 results
- Build the response
Strengths

- Both simple word and phrase matching
- Ability to provide surrounding text to put in context
- A potential good start for a more generic Content Search API implementation?
Limitations

- Tightly coupled to Stanford’s infrastructure
- Can’t effectively phrase search across pages/canvases (no multi-hit annotations)
- Only intended for searching OCR text
- No support for motivation, date, and user queries
- No persistent or dereferenceable annotations, even for text resources
Challenges

- Identifying things beyond scope of the specification that are supported (e.g. phrase searching)
- Authentication for restricted text resources
- Client behavior
  - UV expects Content Search API 0.9 responses
  - Adding support for multiword autocomplete: [https://github.com/UniversalViewer/universalviewer/pull/552](https://github.com/UniversalViewer/universalviewer/pull/552)
- User experience and import to viewer behavior
Thank you!

Mark A. Matienzo
@anarchivist
matienzo.org/presentations
2018 IIIF Conference
24 May 2018

Virtual Tribunals Team
- Cathy Aster
- Chris Beer
- Gary Geisler
- Darren Hardy
- Jessie Keck
- Kris Kasianovitz
- Mark Matienzo
- Jack Reed
- Penelope Van Tuyl
- Camille Villa
- Drew Winget