Presenting Augmented Reality Component for review Part 2

Before we clarify what is, and why, a basic Augmented Reality functionality, some clarification of terminology should be in order. I use term Abstract Reality, or sometimes Abstract worlds, as a superset for the following technologies: both Virtual and Augmented reality, as well as for Holograms and Direct Neural Interface. Of those, I am convinced DNI will prove to be the most impactfull in our future. I mean, it is too easy to imagine managing robots that mine near celestial bodies, or repair and build structures in dangerous or inaccessible places like ocean's depths, from the safety and comfort of one's work place.
It is a distinction that helps me clarify my thoughts. If I use it, you will know meaning I am trying to convey.
Another distinction I am prone to use is using peer-review software, or more generally, peer-review IT technologies instead of more customary expressions open or free software.
Why?
It simply seems more to the point, and therefore helpful. Even worst ignoramuses know that peer-review is crucial part of Scientific Method, and some, even manage to comprehend why is it so. Thus, importance of open-source technologies doesn't have to be explained, for the n-th time. Basically, I've found out that it saves me time.
Hence, if I use it here, you will know what I am trying to say.
So… what is Augmented Reality's basic functionality in within the MIT App Inventor context? After giving some thought to it, I've concluded that providing features for custom Camera Applications is appropriate start. That assumes, in my opinion, access to Camera preview, functionality to save it in desired location, alone, or together with an camera overlay. Which is to say, capability to construct camera overlay of choice is only limited with other MIT App Inventor features. Of course, when one starts to consider other features MIT App Inventor provides, mind boggles. Therefore, let us, briefly, take a look under the "hub", metaphorically speaking.
Relative Layout is the main, or root, layout, for ARC, immediately on top of it is a TextureView that "holds" camera preview, and on top of it can be whatever overlay one thinks of in the MIT App Inventor.
Those knowledgeable in the subject can immediately discern some, if not all features, I work on, and hopefully will add in the future.
However, problems are many. Documentation when exists, is misleading or plain wrong. For instance, it seems that achieving transparency (i.e. seeing through it) of camera preview is impossible, despite "camera transparency method".
Which was a bad news for me. I had an idea, inspired by numerous "combined" photographs one can find on the net, where one can see today buildings or scenery, on top of which were people or events from past times.
It seemed to me that it would be a nice asset for both historical education, art, entertainment and tourism, to name just those. For instance, imagine going to Paris and being able to see on a camera preview how 1900 Olympic Games looked like, or go to Crete and look at the latest reconstruction of Minoan world… Possibilities are endless, especially when one realizes that nothing, nothing but imagination, prevents modern artists and archaeologists to offer their own visions how life may looked like at the time, or at least inspiring clever fiction rooted in facts.
I know, similar effects can be made in other ways, still, it is a feature I hope I will find a way to implement somehow.
It would be too long to describe all the ways I tried to implement that, those interested, please contact me.
Camera rotation, it seems, is something every manufacturer has its own idea what it is, so I was eventually forced to drop it out, even lock the camera in portrait, for this alpha version of ARC. It simply is not efficient, it often freezes application, or causes a loss of camera preview on many devices. I hope someone will be able to point me in the right direction, or devise a solution.

Same, or similar, can be said for camera effects. As far as I was able to discern, there is no consensus what should be minimally available camera effects in Android, it is left for manufacturers to decide. Which prompts me to ask, do I want too much?
Am I mistaken with the approach of cramming too many features into one component? I mean, it seemed redundant to provide video saving, when we already have a component that does precisely that, and I didn't succeed, yet, in saving camera video and its overlay on SD card.
Maybe it would be wisest to offer several versions of ARC controls? For instance, one for the basic functionality we can be sure will work on the minimal MIT App Inventor requirements, second one with all the features we can implement and think of, and perhaps, the third one, that will exclusively use Androids Camera2 API?
I do not know. I am not smart enough, advice please?
To continue, I tried keeping code as straightforward as possible, while keeping it in line with MIT App Inventor code conventions, as I was able to comprehend them. I even reused code whenever I could, for instance, logic for saving pictures, with overlay or not, is from the Canvas component, only a slightly adapted to suit new environment.
I made it a non-visual one, primarily in order to offer it as an extension so that as many as possible people can use it immediately, but a welcome consequence is that it also helps ordinary users to use more straightforwardly, in my opinion.
Which is to say, in design tab, they put it as they would any non-visual control, and in the blocks part of App Inventor, they add their design to "AddPerspective" method. I do it immediately in the screen initialization event, but I am sure, other people will think of other ways, please let me know what are results.
Naturally, for, let's say, completeness sake I also added "RemovePerspective" method, though one can see how frequent addition, and removal of Abstract Realities perspectives could lead to problems for hardware platforms.
As for the name "perspective" or more verbosely "Abstract Reality perspective", it seemed appropriate to me. Why? Because we may start thinking, thanks to all the features MIT App Inventor offers, that reality is becoming yet another medium for our creative expression. Hence, word perspective seemed fitting one to use.
Of course, when I said reality in the previous paragraph, I was to the referring one we all share, one where all hardware infrastructure is; also the one with (still) best graphics, but also the one with the lousy script.
I also wanted to implement seamless camera change, from back to front and vice versa, but the problem is architectural in nature. In order for ARC to implement that, and still be a viable extension, support for that should, to the best of my knowledge, be in the same class. Which implies implementation of said functionality through property change support, at least, as far as I know. However, I noticed, in MIT App Inventor listeners are used, of course, necessary listener, I imagine, could be implemented as a subclass, but it strikes me as inelegant.
Other possibility, it seems to me, is to use new Google library for Data Binding. But how does it play with various Android version? I've glanced it, while I was researching this problem and the one for seamless manipulation of camera preview size, and impression I got is that library is exclusively for Android 2, onward. Am I mistaken?
To add more to confusion, I also read that best way for "contacting" Android UI thread is to use handlers…. Advice anyone?
Maybe the wisest approach would be to provide some general purpose listener for property change, in the MIT App Inventor, which could then be used for various purposes. If there is already such a thing, please excuse my ignorance and let me know of it.
I am sure there is much more to be said, but at the moment, nothing else comes to my mind, so let us start discussing intriguing possibilities ARC with MIT App Inventor offers to us.
Though I touched previously on one trivial educational possibility, thanks to the success of Pokemon Go, I have no doubt majority of early efforts would go along those lines, people trying to offer new capabilities, or make their own clone. Sooner or later they will realize two things:

First one is, there is a lot of fictional worlds to choose from beside Pokemon, and the second one is, there are also mythologies and mythological beings one can choose to bring to “life” without a fear for burdening oneself with displeasing intellectual properties cases. Who knows, maybe there is already among us a rebellious soul prepared to offer Vatican some type of exclusivity to certain AR? Hmm…

As I’ve implied before, we might as well start understanding reality as a medium for creative expressiveness, so what about street art? One can easily imagine graffiti, welcome or not, “somewhere” in Abstract Realities sharing camera view with certain places. Why stop there? Today audience on concerts, plays and any other events tend to use smartphones, you know, to postpone actual enjoyment for later, so why not count with that? Role of directors could be augmented even further with these technological amenities.

All of those, and many other, we may start considering as a type of interactive experience. Can we stretch the definition of interactive/experimental/street theater to encompass such events? Enriched with AR technology? I am curios to see, hopefully as soon as possible, street art and theater along this lines.

More personal experience could be a Dungeon Master that took some time to embellish the story for its players with complementary AR artifacts. Or a brief adventure game with the charters from the newest movies, and experience that could go hand in hand with more traditional movie trailers. Or maybe, people casting spells, reliving Harry Potter, for instance, in the actual world.

Come to think about last example, we already can purchase keychains we can than track with our smartphones in the case we misplaced them, so what is stopping people to put them on sticks and call them wands? How feasible is that? Or any of the previous ideas?

I do not know.

I hope more knowledgeable about corresponding components (lets see, Bluetooth, GPS, sensors, is that all?) will find time to clarify that matter. Maybe the smartest way will be to offer a component specialized for the placement, and thus, recognition of placement, of AR artifacts?

I do not know. Clarification, advices, insights etc, are welcome and appreciated.

Those are just some of the trivial examples that come to my mind, if you disagree with that notion, just remember that I had some more time to familiar myself with those concepts, and that is all.

All in all, when I now think about them, I can not shake the feeling what a misstep in marking was Google Glasses. Do they really thought that main selling point should be effortless spying on people, when one, for instance, commutes? I will not presume what other people find interesting, self-embarrassment on social media certainly has its devout audience, but I do not know that in the morning (in big cities, for example), one is not inclined to connect with people one probably will never see again. One strives to get to the places comforting in their predictability. I think I once read that we are actually biologically limited how many persons we can process during the day, and we protect ourselves from that particular mental overflow in various ways, habits are highly useful in that context. Imagine the horror if anyone you will meet during the day while commuting or working, imposes their whole existence on you, instead of simply remaining a more or less polite stereotype.

On the other hand, there could be something like this:

Tired of the world? Displeased with it?

Well, why not make your own?

How?

MIT App Inventor supported Augmented Reality, of course!

Catchy?

Whatever your answer may be, I think we can all agree that world is better off without me in the marketing.

To generalize, problems with the above use-case scenarios can be, as far as I am able to comprehend, put in three categories. Problems of visual representation (visibility, for short), problems of AR artifacts placements, and problems of connectivity (real-life storage of artifacts, their retrieval and search, and communication with the user).

Of those, you can now experiment for yourself how are we stand regarding problems of visibility. Problems of placements are, in my opinion, solved long ago, or will be solved in certainty. Essentially, all we need to solve them is to choose appropriate Mathematical tool for a particular problem at hand.
By far, and I had some time to examine the matter, I am most worried with the connectivity problems. In the age of continuously improved cloud services and internet technologies, it may seem as a stretch, but various pitfalls will soon became apparent.
Unfortunately, I've planned to use GCS for some basic experimenting regarding those problems, and share results immediately, but that is now postponed indefinitely.
Still, there are at least two good news. One is MIT App Inventor has in itself a scheme, a language of high abstract expressiveness (highest in my opinion, mathematically speaking, I would add, but that is entirely different subject, and it is one of the main reasons why I become interested in App Inventor, I hope I will be able to talk about about role of the scheme in App Inventor with someone knowledgeable).
The other one is, hopefully inevitable, role of peer-reviewed IT technologies example of are arduino and raspberry pi. I mean, if we already have technology we can use for tracking movement of sticks (presumably), what is stopping us to wear those simple transceivers in jewelry or cloth? I am certain that many would appreciate possibility to look to others as their favorite fictional character, or like their avatar in some virtual experience (video games is such a laughable term today, in my opinion)? After doing that, people will become more comfortable with wearable mini computers performing other tasks.
Who knows, maybe one day very soon some smart people will actually build a support for wearable (or at least, portable), mini computers that together can serve a bit of hardware like Google Glass were, and it might become a beginning of something new.
Why should they do that?
Strive for creation, curiosity, wish to leave one's impact on the future are all good answers, but one more down to earth is, it isn't all about fun and games.
And Education I feel compelled to add, applicability of ARC on education is my current focus actually. My first choice of demo application was to make a puzzle game based on Mathematical proofs and discoveries, and then adventure-like game where one must provide answers to solved scientific problems in order to move on. But then I realized it wouldn't be appealing to most people, and I didn't find a way to cram desired functionality that can be illustrative of technological capabilities. It does beat senseless sitting in class memorizing vapid, dry facts, in my opinion.
What I am actually aiming at can be a good homework for familiarizing oneself with ARC. I already provided infrastructure for it in the examples.
Instead of wigs, I mean haircuts, put furniture, or cloth, or colors as image resources, and that is essentially it. Now move around around your apartment and see for yourself how would they look in your apartment. Or if you have chosen cloth, ask your friend to take a picture of you with it.
Actually, I intended to provide similar application as an example, having in mind how helpful that may be to craftsmans and other delightful creative people, but I settled on haircut/drawing on one's appearance, thinking it will be more interesting to greater number of people.
That aside, you just made android application that could allow online providers like Amazon to achieve another type of penetration in the market. Consider, most of people are reluctant to buy certain products (or any) online, simply because they can not see for themselves how would they fit in their apartment, or on them. Now that obstacle is no more.
Sure, there are a lot of conceivable issues, for instance, dimensions, but those are solvable, no reputable online salesman, or one that aspires to become such, would dare, hopefully, to cheat on measurements, by providing misleading image resources.
Of course, you can tweak your application via App Inventor to your heart content, and I hope you would. I wouldn't mind to be delighted with novel idea, quite the contrary.
Let's spend a few words about about other changes one can find in ARC fork on github.

They all are easily if one knows that for some time now, I am interested in type of problems I come to call problems of abstract expressiveness. It is quite a grandiose term for a common sense notion, which is: we can not solve a problem we are not able to conceptualize. Consequently, better we are in representing problems, better we should be in solving them.
What that has to do with anything?
Well, it seems, that there is a Mathematical answer what is a meaningful conceptualization of a problem, which includes user interface. Mind, it doesn't necessarily follow that such interface is also pleasing to look at. Ergo, it explains my fooling around with button shapes, some more or less ill-conceived additions to Canvas.
I also added margins and padding to Visual Components, in a way I think, is best for majority of users. Meaning, in design type with one number they can set all margins and/paddings, but if they need to be more precise, then they will have to use blocks.
I've changed installation to prefer external SD card. Why? It simply seemed more considerate to me. Not all kids interested in learning programming can afford hardware of necessary capabilities. Actually, if they can, chances are they are not interested in programming at all.
However, for duration of the class, they may be lend adequate smartphone, and then they simply put in it their own personal SD card, and that is that.
I also provided annotation for UsesFeature, which is, I guess, self-understandable when one works with the camera. However, thanks to my inexperience with git, some unnecessary conflicts arose, so I omitted support for it. They are still traces of it in code, though.
Oh yes, few times I noticed that if overlay is 100% percent, than it doesn't show on camera preview at all. I think I solved that problem during one of my numerous architectural refactoring, but do let me know if it arises again.
I would like to apologize for not following recommended work-flow for suggesting a MIT App inventor component, thing is, I didn't know will I have time to finish it, so I worked on it, when my other obligations permitted, and then it just sort of happened.
Well, that is all for now, there are a lot of other things I would like to, at least, touch upon, In fact, I feel like I merely glanced upon the surface, but it will have to wait.
Respectfully,
Tomislav Tomsic